applied sciences

MDPI

Review

Survey on Quality of Experience Evaluation for Cloud-Based Interactive Applications

Jesus Arellano-Uson [1,*], Eduardo Magaña [1,2], Daniel Morato [1,2] and Mikel Izal [1,2]

[1] Department of Electrical, Electronic and Communications Engineering, Public University of Navarre, 31006 Pamplona, Spain; eduardo.magana@unavarra.es (E.M.); daniel.morato@unavarra.es (D.M.); mikel.izal@unavarra.es (M.I.)

[2] Institute of Smart Cities, Calle Tajonar 22, 31006 Pamplona, Spain

* Correspondence: jesus.arellano@unavarra.es

Abstract: A cloud-based interactive application (CIA) is an application running in the cloud with stringent interactivity requirements, such as remote desktop and cloud gaming. These services have experienced a surge in usage, primarily due to the adoption of new remote work practices during the pandemic and the emergence of entertainment schemes similar to cloud gaming platforms. Evaluating the quality of experience (QoE) in these applications requires specific metrics, including interactivity time, responsiveness, and the assessment of video- and audio-quality degradation. Despite existing studies that evaluate QoE and compare features of general cloud applications, systematic research into QoE for CIAs is lacking. Previous surveys often narrow their focus, overlooking a comprehensive assessment. They touch on QoE in broader contexts but fall short in detailed metric analysis. Some emphasise areas like mobile cloud computing, omitting CIA-specific nuances. This paper offers a comprehensive survey of QoE measurement techniques in CIAs, providing a taxonomy of input metrics, strategies, and evaluation architectures. State-of-the-art proposals are assessed, enabling a comparative analysis of their strengths and weaknesses and identifying future research directions.

Keywords: cloud-based interactive application; remote desktop; cloud gaming; QoE; interactivity time

check for updates

Citation: Arellano-Uson, J.; Magaña, E.; Morato, D.; Izal, M. Survey on Quality of Experience Evaluation for Cloud-Based Interactive Applications. *Appl. Sci.* **2024**, *14*, 1987. https://doi.org/10.3390/app14051987

Academic Editor: Giacomo Fiumara

Received: 11 January 2024
Revised: 9 February 2024
Accepted: 26 February 2024
Published: 28 February 2024

1. Introduction

Cloud computing is a growing computing paradigm [1] that promises on-demand computing resources. It is possible to differentiate between three types of cloud deployments: public clouds, private clouds, and hybrid clouds. In the public cloud, an off-site third-party provider offers computing resources and management. These resources are accessed through the Internet. In this paradigm, there are new billing schemes whereby customers pay for the use of the infrastructure. Users can access high-power computational resources from client machines, with the only requirement being a connection to the Internet. Thus, individuals and companies can reduce the cost of deploying and maintaining their computing infrastructures by delegating system maintenance to the cloud service provider, which offers infrastructures with high bandwidth and low latency [2].

In contrast, the private cloud refers to computing resources within a private network. Private clouds are for the exclusive use of a single customer, and the customer's service administrators manage computational resources. On the other hand, hybrid cloud computing offers an infrastructure in which public and private clouds communicate with each other, sharing data and applications between them. Although the use of private clouds has been widespread within the industry, the migration from private clouds to hybrid or public clouds is booming. In particular, the costs associated with the public cloud and the data protection policies of each country make the hybrid cloud option attractive [3].

In this article, we focus on cloud-based interactive applications(CIAs) [4], also known as cloud-based distributed interactive applications (CDIAs) [5] or real-time interactive applications (RIAs) [6]. CIAs are applications running in the cloud with strict interactivity requirements. Users expect a near real-time response to their actions, which occur through keystrokes or mouse clicks. The response leads to a screen update so fast that the user cannot perceive that the application is not running locally. CIAs can be deployed on any type of cloud. The existing literature identifies three types of CIAs: remote desktop services, cloud gaming services, and interactive web applications [5]. Services such as video on demand or voice over IP are not considered CIAs because they do not involve interactions as defined above. They use streams of audio/video data with temporal requirements but without requirements about how the user interacts with the service.

The International Telecommunication Union (ITU-T) defines quality of experience (QoE) as "the overall acceptability of an application or service, as perceived subjectively by the end-user" [7]. Laghari et al. defined QoE as "a blueprint of all human subjective and objective quality needs and experiences arising from the interaction of a person with technology and with business entities in a particular context" [8]. One of the main network metrics related to the QoE in CIAs is the time elapsed from when the user interacts with the application until receiving a graphical response. This metric is called the interactivity time [9] or responsiveness [10]. Increased interactivity time can reduce the usability of a CIA. In traditional desktop applications or local services not deployed in the cloud, interactivity time is important. However, in the case of CIAs, servers can be located further away from the user, thus increasing interactivity time, at least on a scale directly proportional to the round-trip time (RTT). In addition, the sharing of remote servers can lead to slow applications. These services can be considered elastic because the perception of interactivity time is gradual and influenced by network and service conditions [11]. Other metrics that influence the QoE are the quality of the video image and, to a lesser extent, audio quality.

Ensuring quality of experience (QoE) in interactive services is of great interest to service providers. For example, an increase of 500 ms in interactivity time for interactive web services can result in significant costs and reduced user activity [12]. The growing deployment of cloud services, especially those with interactivity requirements, highlights the need to quantify and monitor the QoE.

The COVID-19 pandemic has enforced measures in favour of public health, requiring the facilitation of remote work for a significant portion of the workforce. Many employers and employees, who had not previously participated in remote work arrangements, have transitioned to this new model. Studies in the literature [13] have suggested that approximately 40% of large and small companies expect that 40% or more of their workers who switched to remote work would continue to do so even after the health crisis. These estimates indicate that at least 16% of workers will perform their jobs from home at least two days a week as a result of the COVID-19 pandemic. A significant number of services enabling remote work fall within the domain of CIAs [14–16]. Ranging from office automation to remote desktops, these services are increasingly prevalent and require the development of precise ad hoc monitoring to ensure the necessary QoE [17]. Remote employees relying on services like remote desktops benefit from a seamless experience and good interactivity, resulting in fewer complaints for the support centre and allowing end users to concentrate on their tasks. Companies are actively working to guarantee a high QoE for these services, ultimately ensuring the productivity and satisfaction of their remote employees [18,19].

Given this scenario, it is essential to measure the QoE of CIA users in deployments that are increasingly complex. In this work, we conduct an evaluation of existing state-of-the-art proposals for assessing QoE in CIAs. We describe the analytical tools and identify three stages common to most proposals in the state of the art: input, processing, and output. We also compile the metrics used by the tools and describe how they are transformed throughout these three stages.

Our study makes a significant contribution by focusing on strategies for monitoring QoE in CIAs and proposing improvements. To the best of our knowledge, there are no surveys on QoE in CIAs, although related survey papers are presented in the relevant sections. After identifying the challenge of ensuring productivity in CIAs in the post-COVID-19 context, it becomes evident that studying existing proposals for QoE assessment is crucial. We analyse over 28 proposals, identifying key aspects of human perception. Additionally, we explore the architecture and operation of CIAs, distinguishing them from other applications in the non-interactive cloud. We classify the proposals into five main categories based on their strategies and group metrics. Finally, we dedicate a specific section to open issues, where we discuss possible avenues for improving QoE measurement in CIAs.

The remainder of this paper is structured as shown in Figure 1. Section 2 reviews related works and addresses human perceptions of users and their influence on the CIA's QoE. Section 3 presents the categories of CIAs and how the state-of-the-art proposals focus on each of them. Section 4 details the characteristics of the CIA's infrastructure and the different location possibilities for QoE monitoring tools. Section 5 groups the literature proposals into the main strategies used to measure QoE in CIAs. In this section, we detail how each proposal derives its measure of QoE. Section 6 outlines the three stages these strategies must go through for QoE assessment. Finally, Section 7 explores open issues, and Section 8 concludes this paper.

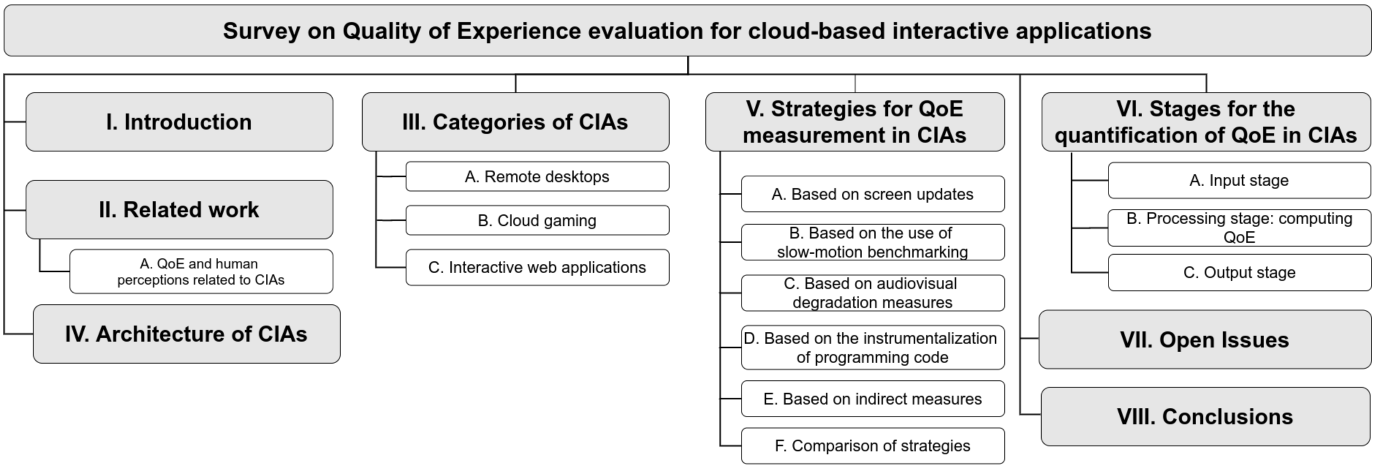
Survey on Quality of Experience evaluation for cloud-based interactive applications
I. Introduction
II. Related work
A. QoE and human perceptions related to CIAs
IV. Architecture of CIAs
III. Categories of CIAs
A. Remote desktops
B. Cloud gaming
C. Interactive web applications
V. Strategies for QoE measurement in CIAs
A. Based on screen updates
B. Based on the use of slow-motion benchmarking
C. Based on audiovisual degradation measures
D. Based on the instrumentalization of programming code
E. Based on indirect measures
F. Comparison of strategies
VI. Stages for the quantification of QoE in CIAs
A. Input stage
B. Processing stage: computing QoE
C. Output stage
VII. Open Issues
VIII. Conclusions

Figure 1. Structure of this survey.

2. Related Works

In this survey, we focus on identifying the strategies that can be used to achieve adequate QoE monitoring in CIAs and explore new possibilities to improve current proposals. To the best of our knowledge, no previous surveys about QoE in CIAs exist because of the relative novelty of this type of deployment and the rapidly changing landscape of cloud technologies. There are only surveys with limited scopes related to QoS in cloud computing, interactive applications, general QoE, or QoE for specific services.

Some studies in the literature analysed aspects of cloud computing, such as pricing, scalability, and architecture. Among the surveys that addressed generic QoE, some were barely descriptive and focused mainly on aspects such as architecture and provisioning [20] but they were not specific to CIAs. Barakabitze et al. [20] attempted to provide a tutorial and survey of QoE management solutions in general multimedia services. However, the paper skipped over QoE quantification. It did not consider the particularities of a CIA and only devoted two paragraphs to QoE measurement. Furthermore, the authors only listed two QoE quantification proposals focused on obtaining QoS-specific network parameters, without explaining the procedures the proposals follow.

Other proposals addressed the evaluation of QoE quantification methodologies but with a strong focus on a particular service. Within this group, some surveys focused on a specific field, such as mobile cloud computing [21,22] or multimedia services [23,24]. Shakarami et al. [21] reviewed the mechanisms of offloading computation in mobile cloud computing environments. However, they only dedicated one paragraph to the idea that service administrators should monitor its QoE. They provided a list of works dedicated to assessing QoE in mobile cloud computing. However, the authors did not describe these proposals, nor did they mention what metrics they provided or used, or to what type of services they applied. Skorin-kapov et al. [23] discussed concepts related to QoE management of networked multimedia services and addressed the implications of the emergence of new services. The authors outlined some proposals for QoE modelling. However, the scope of the paper was very broad and encompassed proposals for any type of multimedia service, regardless of whether the services analysed were CIAs or not, thus ignoring their particularities. Laghari et al. [22] provided a recent review of quality of experience (QoE) in the broader domain of cloud computing. The study compiled recent articles from the past few years but did not focus specifically on interactive applications deployed in the cloud. It did not address the specificities of these services, as it evaluated only a few proposals related to cloud gaming. The authors did not delve into a detailed analysis of the evaluated proposals, such as the input metrics required for processing. However, they emphasised the QoE metrics provided in the literature.

Min et al. [24] provided an extensive review of contemporary signal-based audio and video perceptual quality assessment methods. The authors conducted experiments utilising subjective quality scores to analyse and compare the efficacy of these methods. While the metrics they investigated hold promise for evaluating two crucial dimensions of QoE in CIAs—visual and audio quality—they notably omitted state-of-the-art proposals tailored specifically to CIAs. Instead, their focus remained exclusively on visual and audio aspects applicable to any service, irrespective of its interactivity or deployment in the cloud. Additionally, for the majority of the metrics assessed, the direct relationship with QoE was not delineated, with the analysis predominantly confined to QoS considerations.

Other surveys focused exclusively on remote processing or cloud gaming [25–29]. Cai et al. [25] analysed the latest research on cloud gaming from different aspects, covering cloud gaming platforms, optimisation techniques, and commercial cloud gaming services. Within the paper, they devoted a subsection to analysing proposals for evaluating QoE in cloud gaming but overlooked other types of CIAs. However, the authors provided an enumeration of the content of the works without addressing the metrics or processing details. The authors did not provide a taxonomy of the proposals or a comparison between their advantages and disadvantages. Laghari et al. [26] analysed the cloud gaming environment and looked at future development issues that could help provide QoS in line with service-level agreements and increase the satisfaction level of cloud gaming users, thus improving QoE. In their work, they only dedicated a subsection to exposing methods for QoE quantification in cloud gaming, ignoring other types of CIAs. They grouped together proposals that evaluated video streaming, proposals that evaluated network aspects, and proposals based on their qualitative comparison of characteristics and performance. However, the number of proposals analysed was small, and the authors did not emphasise the difference between QoS and QoE metrics. Shi et al. [27] reviewed interactive remote rendering systems proposed in the literature. They provided a tutorial on how these systems work (architecture, rendering, examples, and challenges). In addition to not taking into account other types of CIAs, the authors only dedicated a small subsection to explaining the importance of evaluating QoE. However, it was a summary of proposals; they did not describe each work, the metrics, or the processing used. Moreover, they did not provide a taxonomy or compare the advantages of each of the proposals. None of these papers covered the three main categories of CIAs: remote desktop, cloud gaming, and interactive web applications.

Other works focused on a specific CIA of recent interest, such as cloud gaming. This is the case with Metzger et al. [28] and Abdallah et al. [29]. In these works, the exclusive focus on cloud gaming prevented the leveraging of strategies employed by other authors in other CIAs for QoE quantification. In the case of Metzger et al. [28], they conducted a survey outlining the architectural peculiarities and user requirements of these types of services. However, the processing of some proposals was not detailed, making it difficult to understand the derivation of QoE metrics. Abdallah et al. [29] concentrated on examining QoE provisioning concerning delay-sensitive video computing requirements, encompassing various facets such as service ubiquity, device diversity, network heterogeneity, and operational variability. The paper delved into the challenges associated with QoS provisioning within converged networks and explored their ramifications for QoE provisioning, with a particular emphasis on the realm of cloud gaming. Despite extensively reviewing and analysing relevant literature, the paper fell short of presenting a taxonomy elucidating the parameters, processing methodologies, or output metrics employed.

The closest studies to our work are those carried out by Wei Tang et al. [30] and Casas et al. [31]. In their survey, Tang et al. [30] studied proposals for QoE evaluation of mobile cloud computing. However, they focused exclusively on evaluating services that can be accessed from a mobile device and overlooked any other types of CIAs. They only addressed the quantification of responsiveness and visual content quality. The number of proposals was small, and there was no comparison between them. They did not clarify the difference between the proposals and their advantages and drawbacks. The survey by Casas et al. [31] mainly consisted of a proposal for quantifying QoE in cloud services. However, the authors carried out a prior analysis of the state of the art to describe other proposals. The survey focused on general cloud services and not CIAs, so most of the proposals ignored their particularities. Although the authors attempted to describe the proposals, they did not describe the metrics or processing used, nor did they provide a comparative taxonomy of the proposals.

Table 1 outlines the contributions and shortcomings we have identified in previous surveys and our proposal. The table highlights whether the surveys take into account the relevance of QoE in particular CIAs (remote desktop, cloud gaming, or interactive web applications) or, conversely, if they focus on general cloud services. It further classifies the surveys based on the context they provide for understanding the QoE domain of CIAs. The table also identifies which surveys provide information on architectures and user perceptions relevant for QoE evaluation, as well as those that differentiate between QoS and QoE measures. We identify the surveys that clearly address the quantification of QoE. Table 1 also identifies the surveys that offer an analytical taxonomy of the proposals, highlighting the details the taxonomy takes into account to be able to replicate its implementation (input, processing, and output metrics). Finally, the table shows the number of papers cited by each survey focusing on QoE monitoring and measurement.

QoE and Human Perceptions Related to CIAs

The stimuli perceived by the user influence the quality of experience provided by a CIA. In this section, we discuss the human perceptions considered crucial in the state of the art for quantifying QoE in CIAs. In the literature, these human perceptions, also referred to as physiological aspects, play a significant role in shaping the user experience [28]. Three primary human perceptions have been identified: visual quality, audio quality, and interactivity or responsiveness. In most digital systems, audio and video signals are fundamental components [24]. Subjective evaluations of audiovisual quality are widely regarded as the most accurate method to reflect human perception [32]. However, given the idiosyncrasies of the applications under review in this study (interactive applications), the tools and methodologies for QoE quantification also require metrics that evaluate the strict temporal requirements between user interactions and system responses.

Visual quality encompasses the aspects of an image perceived by the human perceptual system, significantly influencing the end-user experience. Content may experience degradation, such as a decrease in the quality of textures in a video game, pixelation of the remote desktop screen, or missing video frames, compared to the original or expected content. This degradation can impact visual quality. Researchers have proposed various metrics to quantify these degradation artefacts.

Table 1. Asummary of related surveys and systematic discussion.

	Target			Scope			Taxonomic Analysis			No. of QoE Papers Reviewed
	Remote Desktop	Cloud Gaming	Interactive Web Applications	Architecture	Human Perceptions	QoE	Input Metrics	Processing	Output Metrics	
Tang et al. (2014) [30]	×	×	×	×	×	✓	×	×	×	7
Casas et al. (2014) [31]	✓	×	×	×	×	✓	×	×	×	16
Shi et al. (2015) [27]	✓	✓	×	✓	×	×	×	×	×	9
Cai et al. (2016) [25]	×	✓	×	✓	×	✓	×	×	×	12
skorin-kapov et al. (2018) [23]	×	×	×	×	✓	✓	×	✓	×	28
Abdallah et al. (2018) [29]	✓	✓	×	✓	×	✓	×	×	✓	21
Laghari et al. (2019) [26]	×	✓	×	×	×	×	×	✓	×	6
Barakabitze et al. (2020) [20]	×	×	×	✓	×	×	×	×	×	2
Shakarami et al. (2020) [21]	×	×	×	✓	×	✓	×	×	×	15
Min et al. (2020) [24]	×	×	×	×	✓	×	✓	✓	✓	12
Metzger et al. (2022) [28]	×	✓	×	✓	✓	✓	✓	×	✓	9
Laghari et al. (2023) [22]	×	✓	×	×	×	✓	×	✓	✓	5
Our survey	✓	✓	✓	✓	✓	✓	✓	✓	✓	28

A second human perception that the tools evaluate is the audio quality of the services. Equivalent to video quality, audio quality is understood as the aspects of the audio signal perceived by the human auditory system, which impact the final user experience. Similar to visual quality, audio quality can be impaired by the degradation of the sonorous content compared to the expected or original content (e.g., clarity of the sounds of a video game hosted in the cloud). In this area, we present the efforts made to obtain metrics that quantify the quality or degradation of the audio.

Audio and visual quality have varying degrees of relevance depending on the service being consumed [24]. This is why the research works studied in this article may emphasise varying degrees of emphasis on the quantification of each perception, depending on the scope of their target services.

Finally, given that the present taxonomy is aimed at CIAs, the third human perception evaluated by the tools and methodologies is the responsiveness or interactivity of the services. The QoE perceived by users is influenced by the time it takes to perceive a response to interactions with the service. This responsiveness can be translated in many ways. There are several temporal references that can influence it, depending on the service or application consumed (e.g., opening time of a window in a remote desktop, deployment time of a field in an online form, agile movement of a player in a cloud gaming service). Again, researchers have made efforts to quantify this interactivity in different applications and services, and we explore them throughout this paper.

The efforts of academia and industry seek to quantify one or more of these human perceptions with particular output metrics depending on the type of CIA.

3. Categories of CIAs

We group the proposals about QoE in CIAs into three main categories. This literature review has revealed that all proposals fall into one or more of these three categories. In this section, we explore these categories and the importance of human perceptions in each of them. Table 2 shows the three categories and the proposals in the state of the art that focus on each (or some) of them. Each of the three categories has different QoE requirements depending on the service consumed by the user. Cloud gaming typically has higher QoE requirements compared to remote desktops, and remote desktops have higher QoE requirements compared to interactive web applications.

Table 2. Papers on the evaluation of QoE in the three types of CIAs.

<table>
<tr><th>Remote Desktops</th><th>Cloud Gaming Services</th><th>Interactive Web Applications</th></tr>
<tr><td>Alali et al. [33], Kumar et al. [34],
Dong et al. in [37], Song et al. [38],
Li et al. [41], Magaña et al. [42],
Exoprise [45],
Casas et al [31],
Nieh et al. [49]</td><td>Liu et al. [35],
Hsu et al. [39],
Shushi [43],
Wang et al. [46],
Graff et al. [48]
Penaherrera-pulla et al. [50]</td><td>Jahromi et al. [36],
Saverimoutou et al. [40],
Hossfeld et al. [44],
Wehner et al. [47],
Casas et al. [31]</td></tr>
<tr><td colspan="2">Nguyen et al. [51]</td><td></td></tr>
<tr><td colspan="3">Varghese et al. [52],
Laghari et al. [53],
Mahmud et al. [54],
Arellano-Usón et al. [9],
Leo Bodnar Electronics [55],
Leo Bodnar Electronics [56],
Johnsen [57],
NVIDIA [58]</td></tr>
</table>

3.1. Remote Desktops

A remote desktop is a CIA that allows users to connect from a local computer to a remote one as if they were sitting in front of the remote computer. The local computer may have low computing resources, and it is called a thin client. The remote computer is usually a virtual machine running on a shared server, although it can also be a dedicated computer. In this CIA, the thin client captures the mouse movements, clicks, and keystrokes to be sent to the remote computer where the application is running. These interactions can prompt screen updates that are then sent back from the remote computer to the thin client. The thin client is only a kind of forwarder of user actions. Remote desktops allow mobility for users (the remote computer can be accessed from any device and at any place) and reduce management and maintenance costs. Some remote desktop applications specialise in performing remote assistance functions, such as TeamViewer [59] and Google Remote Desktop [60]. Other solutions are designed for continuous use, such as Windows Remote Desktop [61], VMware View [62], and Citrix [63]. Remote desktop solutions based on the deployment of remote computers on virtual machines are referred to as virtual desktop infrastructure (VDI) solutions. Amazon Inc. provides this type of infrastructure through a public cloud with the Amazon Workspaces service [64]. The use of remote desktops increased drastically due to the pandemic situation in 2020–2021. For example, in the information technology sector, the increase in remote desktop deployments was 258% in 2022, with more than three-quarters of employees using remote desktops [65,66]. The percentage of companies offering remote work, and therefore, remote desktop solutions, increased from 51% at the start of 2023 to 62% in 2024 [67,68].

In this type of CIA, the user wants to control a remote computer as if it were a local computer, without any noticeable delay or loss in image quality. For example, by clicking to close a window, the user expects the window to close immediately. Two main human perceptions can influence the final QoE. The first is interactivity or responsiveness. The lag between user interactions (keyboard or mouse) and the screen updating as a result of these interactions can affect user experience. For example, clicking a window and moving it around the desktop. The user is accustomed to a local environment. The shared virtualisation server and access network in remote desktops can increase the response time to user actions, with a serious impact on QoE. In most remote desktop services, the server sends the screen updates as it generates them. For example, Sun Ray [69] and Lap Link [70] prepare and send screen updates using this method. In other cases, such as Citrix or Windows Remote Desktop, the server bundles multiple intermediate screen updates to send only a single relevant screen. The client may also send a display update request to the server instead of waiting for a triggered update. The VNC protocol [71] specifies that the server does not send updates until it receives a request for the latest version of the display. Some remote desktop services, such as Lap Link, Windows Remote Desktop, and Citrix, cache part of the screen state locally. Local caches allow for incremental screen updates, taking advantage of regions of the previous screen that could have been moved to other coordinates without receiving an unchanged portion of the screen.

The second human perception that influences QoE in remote desktops is the video quality perceived by the user. Some remote desktop communication solutions, such as VNC and Sun Ray, send the screen corresponding to the user's desktop as a video stream or as a sequence of individual video frames. The video stream is compressed with losses, compromising the perceived visual quality and influencing the final QoE. Other solutions, such as LapLink, Citrix, and Windows Remote Desktop, send graphical screen primitives, such as "plot a window of this size in that position of the screen", to the thin client. The client receives primitives that indicate which regions of the screen must be modified and which content must be represented. The thin client reproduces screen content without any video quality degradation caused by video compression artefacts. However, video quality can be degraded in both alternatives by network packet losses. In addition, the transmission may freeze due to communication problems. This causes stuttering and affects the video quality of the remote desktop. Audio quality is generally less important in remote desktop services, but when needed, the service has to consider the effect of the compression codec, the network packet losses, and synchronisation with the video stream.

3.2. Cloud Gaming

Cloud gaming services are also a category of CIAs. In cloud gaming, users access a remote video game as if it were local. Similar to remote desktops, processing and rendering take place on the remote server, and the user device collects user interactions with the controller, sends them to the server, and displays the resulting screen updates. Using these services, players do not require expensive video consoles with large computational resources. The user device can be a smartphone, tablet, PC, laptop, or even a traditional video console. This flexibility allows the user to play in the cloud from anywhere and resume their games from other devices. In cloud gaming CIAs, there may be increased sensitivity to delay for some types of games. CIAs allow video game developers and publishers to use the cloud resources of cloud gaming platforms. This simplifies their deployment, maintenance, and costs. Additionally, game developers can develop games for a single platform instead of multiple platforms. There are commercial cloud gaming deployments such as Nvidia GeForce NOW [72], Amazon Luna Cloud Gaming [73], PlayStation Now [74], and Xbox Cloud Gaming [75]. The cloud gaming industry is expected to reach a value of USD 3856 million by 2025, which represents an increase of 54.1% from 2019 [76].

The three human perceptions described in Section 2 influence the final QoE perceived by the user of a cloud-based game. The player wants to play a video game as if it were running on their game console and not a remote server. There are two main components of a cloud gaming platform: (1) the game logic, responsible for transforming user inputs into actions, and (2) the scene renderer, responsible for generating the screen updates in real time. A scene in a video game is composed of all the elements present at a specific time and in a specific position. On the client's computer, the command interpreter captures the user interactions that are then sent to the server. The video capturer must capture the scenes of the video game rendered on the server and make a video stream that is later compressed to save bandwidth. Compression influences human perceptions when it is lossy. In multiplayer sessions, each player's game scenes can be used together to perform compression, taking advantage of the redundant information and compressing multiple users' streams together [25]. Additionally, cloud gaming platforms can detect the regions of the game scene of interest to use compression with more bits and, therefore, higher quality [77,78]. Other compression strategies use information from the rendering of video games to estimate movements and save compression time [29]. Graphics compression strategies generate a 3D coordinate space with information on the points of the objects that make up the scene and the 2D texts that cover them [79]. The server compresses this scene and sends it to the client, who must render that information. This strategy is more demanding for the client since it has to perform rendering in the case of the server compressing the video. Lossy compression influences the visual quality perceived by the player and adds a delay to communication.

In gaming, the user expects the service to respond graphically as quickly as possible with good interactivity. The game should quickly execute actions, such as rotating a geometric figure or moving a player, so that the user does not perceive the involvement of a remote cloud platform. The codec directly influences the time spent on compression and, therefore, the perceived interactivity. The cloud server sends an audio stream alongside the video stream. Human perception of audio quality is more important in cloud gaming services than in remote desktop CIAs. Compression, network packet losses, and network packet delays affect audio quality. Furthermore, sound stimuli produced in response to player actions must be presented in a synchronised way.

3.3. Interactive Web Applications

Interactive web applications are also a category of CIAs. They differ from typical web browsing, where each click on a link requires a new webpage download. Instead, interactive web applications refer to single-page web applications that have functionality similar to that of a desktop software application. Users utilise a browser to interact with these applications, relying on JavaScript on the browser side for a substantial portion of the functionality. Communication with the cloud is performed using asynchronous AJAX requests. The server that receives the JavaScript requests must calculate and send the associated response in near real time. Interactive web applications simplify the deployment of applications. The developer does not need to create an application for every operating system, and the user does not have to install different applications on their device. Examples of interactive web applications include Google Docs [80], Office 365 [81], and mapping services like Google Maps.

In this type of CIA, users want to use an application through the browser as if it were a traditional program installed on a desktop computer, without noticeable additional latency. For example, web-based mapping services such as Google Maps or Bing Maps allow users to interactively navigate a map service through a web interface. It is essential that when the user wants to move the map or zoom in on a specific region, the time elapsed between their keyboard or mouse actions and the associated graphical response is as short as possible. Developers of interactive web applications divide the application logic between the client and the server. Interactive web pages consist of four main components: the user interface, client logic, communication, and server logic. The user interface is made up of HTML elements, structured by CSS. The client uses JavaScript to develop part of the CIA's processing logic and capture keyboard or mouse interactions, modifying the structure of the HTML DOM on the fly. Unlike traditional web pages, in these CIAs, screen updates are not generated by a click on a link that opens another static HTML or by periodic JavaScript or CSS updates. Instead, updates are interactively triggered by user interactions. Two main strategies for rendering graphic elements and updating them on the fly are used: modifying the structure of the HTML DOM and interactively drawing on an HTML element called a canvas. The strategy of using the canvas element for graphical rendering is becoming increasingly important for performance reasons. In the canvas element, different graphical elements can be drawn and modified in real time without propagating changes in the structure of the HTML document that can compromise the appearance of the CIA. For example, Google Maps has used the canvas for a long time, and other services, such as Google Docs and Google Sheets, have recently adopted this strategy to improve user interactivity. The client must communicate with the server that executes a significant part of the logic of these CIAs. Web interactive applications normally use AJAX for communication with the server. Some services, such as Google Docs, can generate local graphical updates and wait for the server to confirm them later. However, this is not possible for all types of

CIAs since the service may need active communication with the server side (e.g., Google Maps, where the satellite images are sent by the server).

4. Architecture of CIAs

Researchers and industry are continuously exploring different strategies to quantify QoE in CIAs. Studies obtain different metrics from the infrastructure and evaluate their relevance and the relationship between them. First, we offer a general view of how CIAs work and what differences exist compared to other services and applications. Second, we explore the components of the architecture to understand which ones the researchers focus on.

Unlike local computing systems, the experience perceived by users of cloud services is influenced by the fact that the server executing the applications is not physically located in their vicinity. This requires the dissection of the infrastructure into the components that influence the final service (Figure 2): client, network, and cloud server.

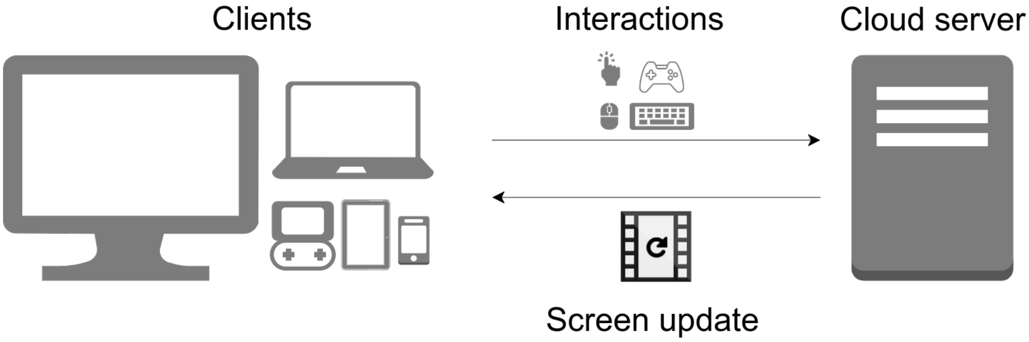

Figure 2. The three components of the architecture of a CIA's services provision.

The client is the first component, and it is the display device with which the user interacts. The client can be a thin client, desktop computer, laptop, game console, smartphone, or tablet. The client has to perform less computational work compared to that carried out by the infrastructure in the cloud since its main purpose is to capture the interactions of the users via the peripherals (keyboard, mouse, gamepad, etc.).

The client sends the interactions to the cloud server through the network (the second component), as shown in Figure 2. The client and the cloud implement a request-response scheme that generates the network traffic to be carried by the network. Some CIAs require high-speed and low-latency networks in addition to the best possible availability.

The last component is the cloud server. The cloud server is in charge of processing client requests and sending the answers back through the network. The responses could be screen updates according to a player's movement in cloud gaming or window movement in remote desktop CIAs. Additionally, they could be the result of a database query or data processed in the cloud server in an interactive web application, prompting a screen update for the client. In CIAs, users interact expecting near-real-time responses. Hence, the server updates the graphical representation on the client so fast that the user does not notice that the application is not running locally.

Figure 2 depicts the common operation of the three types of CIAs. However, the three types of CIAs present differences at a logical level. The common aspect that encompasses remote desktop and cloud gaming CIAs is the fact that the user interface logic is shared between the cloud server and the client's device. The application processing logic, however, is exclusively located on the cloud server, and the client becomes a device oriented to graphical representation and network communication (see Figure 3). In an interactive web application, the client can perform part of the application processing logic without interacting with the cloud server. The bulk of the application logic continues to run on the cloud server. However, the client's browser JavaScript can execute small parts of the service, such as graphical animations of a photographic carousel or simple mathematical operations in a spreadsheet (see Figure 3).

Figure 4 depicts in detail the components involved in CIAs: the client, network, and cloud server. Throughout this survey, we explore the metrics researchers use related to these components, how they use these metrics, and why they chose them.

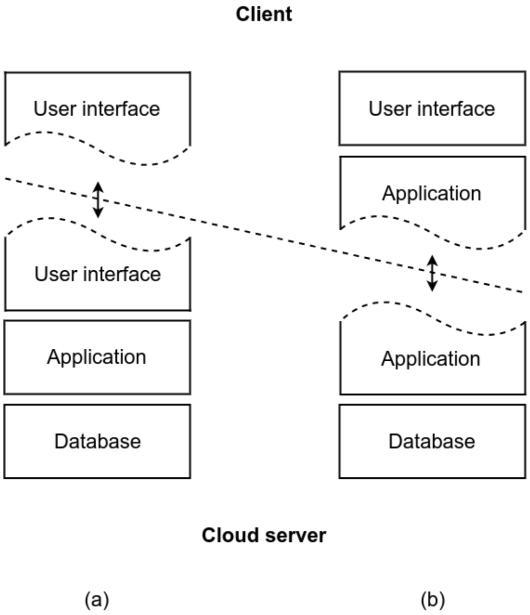

Figure 3. Logical division of CIA's components: (**a**) remote desktops and cloud gaming, (**b**) interactive web applications [82].

On the client side, device usability, usage context, and user expectations and personalities determine the final QoE [30]. The research and tools we analyse seek to identify user insights when using CIAs. However, as direct user opinion is not always available, some measurement strategies obtain performance metrics from the components, evaluating the relationship between these metrics and the QoE. Figure 4 illustrates the three basic subcomponents common to computers, laptops, smartphones, tablets, and video consoles: hardware, operating system (OS), and application. Some QoS evaluation designs obtain measurements directly from the hardware, such as CPU or RAM usage. The OS manages hardware resources and provides services to the applications running on the device. Several research articles, such as Mahmud et al. (2019) [54], Laghari et al. (2018) [53], and Liu et al. (2020) [35], support the idea that the information obtained from the operating system is correlated with user experience. Metrics such as the number of applications or processes running, the number of queued hardware requests, or the screen refresh rate are obtained from the OS. Finally, the application the user is accessing on the client computer (the remote desktop client, cloud gaming client, or web browser) is another component measured using Application Performance Monitoring (APM).

The second component is the communication network. Tools can monitor the network traffic from either the client or the server, offering a network view from one of the communication endpoints. A different strategy is to use a network traffic probe placed in the network path between the client and the cloud server to capture the traffic. The tools process the captured traffic to obtain different metrics: RTT, packet timestamp, data rate, or IP address. Some metrics may require different traffic analysis techniques depending on where the tools capture the traffic.

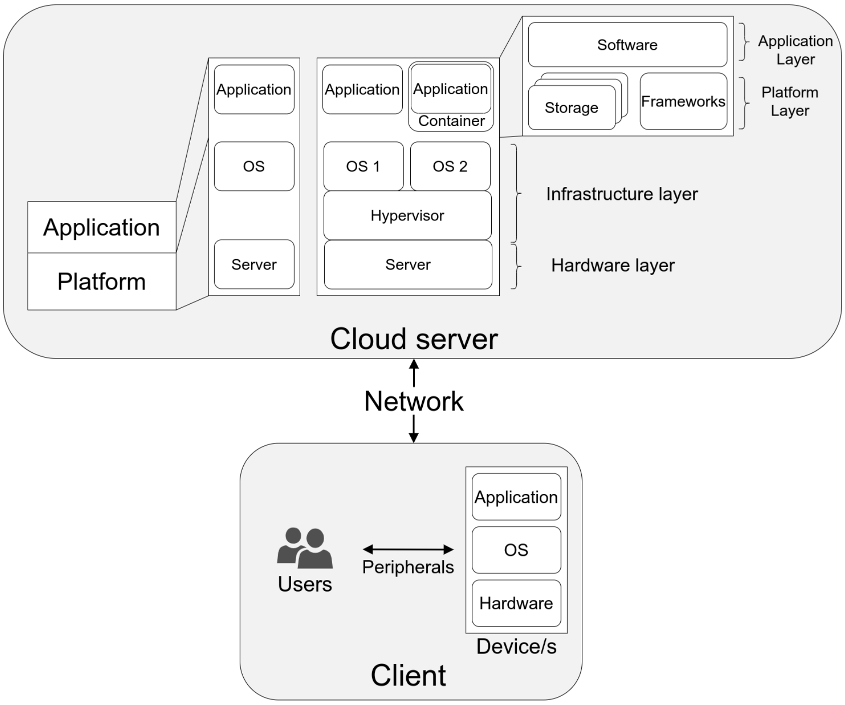

Figure 4. Details of each component of the CIA's architecture.

The third and last component is the cloud server. In [83], the server is divided into four subcomponents: hardware, infrastructure, platform, and application. The hardware layer of the cloud server is similar to the hardware layer of the client, and therefore, the metrics are similar. One of the features that the cloud offers is the flexible use of hardware resources. Cloud server providers select different strategies to take advantage of physical computing resources. They may virtualise the hardware using a hypervisor, running different operating system instances on the same computer at the same time. Service providers may adapt the number of virtualised instances depending on demand. The hypervisor is a component of the architecture that researchers utilise to obtain measurements related to its performance, such as the use of computational resources or the number of virtualised OSs.

The applications run on top of an operating system. Three main subcomponents comprise the applications: software, storage, and framework. The software is the code of the application itself, which is studied using APM techniques. The storage allows the application data to be persistently hosted and queried if necessary. Finally, the framework is an already developed software code that provides structure and functionalities for the development of user-written code. Frameworks are very common in software development and allow standardising their development, reuse, and deployment.

Similar to what happens with an OS, developers virtualise and scale out applications through OS-level virtualisation. In OS-level virtualisation, the kernel allows for the existence of multiple isolated user-space instances or containers. Containers allow application developers to scale out applications in case of high user demand and deploy a new application in seconds, simplifying the deployment process. Similar to the other subcomponents, the containers can be used to obtain QoE metrics such as the number of processes, performance, or the number of applications running.

5. Strategies for QoE Measurement in CIAs

Analysing the proposals in the state of the art, this survey identifies five main strategies for measuring QoE in CIAs: based on screen updates, slow-motion benchmarking, audiovisual degradation measures, the instrumentation of the programming code, and indirect measures. In the following subsections, we explain what each of the five identified strategies consists of, and we explain the particularities of the works in the literature that use them. Table 3 summarises the revised proposals and strategies. Some works combine several strategies in their proposals, but the one considered the main one is identified. For each strategy, we highlight the fundamental procedure and what measures it contributes as a representative metric of QoE.

For the mapping of metrics in a QoE indicator, the literature traditionally opts for the absolute category rating (ACR) [84]. The ACR scale includes scores of "bad", "poor", "fair", "good" and "excellent", applicable to the quality of products or services of any kind. However, there is diversity among the different opinions provided by users [85], which makes it difficult to construct this scale. Each individual who consumes a service or application has different expectations because users are used to a specific environment and may be more or less used to the service depending on previous hours of use. This fact causes researchers to opt for solutions that provide more reliable values than those provided by the ACR scale. To do this, the researchers use averaging techniques to eliminate the noise from influential factors in each user's perception [86]. Averaging user feedback allows us to extract the influential aspects from the set of opinions. The tools use numerical scales to transform qualitative opinions into quantitative ones.

Table 3. QoE quantification proposals grouped by strategy.

Ref.	**Screen Updates**	**Slow-Motion Benchmarking**	**Audiovisual Degradation**	**Instrumentation of Code**	**Indirect Measures**
Nieh et al. (2003) [49]		Yes	Yes		
Shushi et al. (2011) [43]	Yes		Yes		
Casas et al. (2014) [31]	Yes				Yes
Nguyen et al. (2015) [51]		Yes			
Wang et al. (2017) [46]			Yes		
Varghese et al. (2018) [52]	Yes				
Song et al. (2018) [38]			Yes		
Hossfeld et al. (2018) [44]				Yes	
Laghari et al. (2018) [53]					Yes
Magaña et al. (2019) [42]			Yes		
Saverimoutou et al. (2019) [40]				Yes	
Johnsen (2019) [57]	Yes				
Alali et al. (2019) [33]		Yes	Yes		
Mahmud et al. (2019) [54]					Yes
Li et al. (2019) [41]					
Jahromi et al. (2020) [36]				Yes	
Nvidia (2020) [58]	Yes				
Liu et al. (2020) [35]				Yes	Yes
Wehner et al. (2020) [47]					Yes
Leo Bodnar Electronics (2021) [55]	Yes				
Leo Bodnar Electronics (2021) [56]	Yes				
Hsu et al. (2021) [39]	Yes		Yes		
Penaherrera-pulla et al. (2021) [50]			Yes		
Graff et al. (2021) [48]					Yes
Kumar et al. (2021) [34]	Yes				
Arellano-Uson et al. (2021) [9]	Yes				
Dong et al. in [37]			Yes		
Exoprise (2024) [45]	Yes				

Another classic way of measuring QoE within the realm of information and communication technologies is the mean opinion score (MOS) [87], standardised by the ITU. The MOS consists of averaging the subjective evaluations of the users under study under the same service conditions. Currently, the MOS is the standard QoE evaluation measure [88].

5.1. Strategies Based on Screen Updates

Proposals based on the use of screen updates for the quantification of QoE in CIAs seek to evaluate the screen state of the client or server at a given time. They correlate the result of a user's actions with the corresponding result on the screen. For example, if the user clicks on a corner of a window to maximise it using a mouse, the proposal's aim is to measure the time elapsed between the user's click and the screen update reflecting the window being maximised.

Some authors tried to evaluate the time needed to complete a task or a set of tasks. Thus, the authors assume that the less time a user needs to complete these tasks, the better the QoE because the infrastructure provides better interactivity. However, in general, what these proposals have in common is that they do not provide an order of magnitude for the obtained metric to satisfy the user's QoE. In [52], Varghese et al. developed tools to evaluate these elapsed times. To do this, the authors recorded the tasks they wished to study in a database. Each task was characterised by the pattern of keyboard and mouse actions that triggered its execution. In addition, they used the client's screen state to detect any visual elements of interest that should be present at the start and end of the tasks. Thus, they detected the start of a task from the initial patterns of the keyboard, mouse, and screen state (e.g., detecting the icon of a closed folder). Once the authors detected the start of a task, they periodically looked for graphical elements on the screen that indicated its end (e.g., detecting the icon of an open folder) and calculated the elapsed time. To detect that the visual elements of the database were present on the client's screen, in [52], the authors compared pixels one by one, considering only those bitmaps where at least a difference of 35 pixels was present.

In [34], Kumar et al. employed the same strategy of calculating the time it takes a user to complete a set of tasks. To do so, the authors employed the Deskbench tool for quantifying QoE on remote desktops. The difference from the previous work [52] was that the strategy developed by Kumar et al. was not a real-time tool for real users. This tool allows for the emulation and measurement of the time spent performing previously recorded keyboard and mouse actions. During the recording phase, in addition to storing keyboard/mouse information and the corresponding time references in the database, synchronisation points are stored. When the tool replays the recorded actions, the response on the screen can have a significant temporal variability. If the tool attempts to click on an item that the screen has not yet represented, the playback will fail. For this reason, the tool uses the synchronisation points. During the recording phase, the tool obtains synchronisation points every 250 ms, and with them, it stores an MD5 hash containing the areas of interest on the user's screen. During the playback phase of the tasks, every 250 ms, the tool checks the MD5 hash of the user's current screen to ensure that it matches the hash in the database. If it does not match, the tool checks periodically until it finds the expected screen state, and playback can continue. Upon completion of the playback of prerecorded actions, the tool obtains the time needed. In addition, the tool allows for the addition of fuzzy synchronisation points to the database to increase its robustness and not block the playback of tasks due to accidental small differences on the screen.

In terms of commercial solutions, Exoprise's CloudReady tool [45] also opts to record and replay a sequence of actions in remote desktop environments to obtain the time spent on them and thus evaluate interactivity. Unlike the strategy employed by Kumar et al. in [34], CloudReady does not use an MD5 hash to determine that a sequence of tasks is finished but rather uses optical character recognition (OCR) techniques. CloudReady allows the user to specify a text string that the tool locates in a program of the user's choice after opening. The common procedure of OCR systems is to binarize, segment, improve visual quality, and compare with known character patterns. However, as it is a commercial tool, there is not enough information regarding the exact procedure it uses.

Another group of authors has tried to evaluate the elapsed time between when a user interacts with a CIA and the moment they perceive a response to their interactions (this metric is called the interactivity time [9] or responsiveness [10]). In [9], Arellano-Uson et al.

proposed the thin client latency analysis (TeCLA) methodology. The aim of TeCLA is to quantify QoE on remote desktops, independent of the protocol used. The authors measured the interactivity time. To do this, TeCLA periodically (as fast as possible) performs a checksum of the user's screen. When TeCLA detects a keyboard or mouse interaction, it stores the most recent checksum, and it computes the client's screen checksum constantly until it is different from the stored one. At that point, TeCLA interprets that the screen has changed and regards that change as a response to the client's interaction. Then, it calculates the interactivity time as the elapsed time. A change in the screen may not be the actual response to a user's interaction and may generate shorter interactivity times compared to the actual ones. For example, there may be autonomous changes to desktop elements (a clock flashing every second), the current window (a banner containing an animation), or a different window (a video playing on the desktop). The authors call these phenomena instabilities. TeCLA identifies instabilities on the screen when a user is not performing any action. If no instabilities occur for a sufficiently long period, the statistical model of TeCLA interprets the subsequent samples of interactivity as reliable. TeCLA allows for the determination of the minimum interval between the detection of instabilities to reduce the probability of measurement error below a user-specified threshold based on studies with real users.

Instead of obtaining bitmaps from the user's screen, other proposals opted to use external devices. These devices relate variations in light intensity in a particular area of the monitor to the user's keyboard or mouse interactions. When the tool detects a user interaction, it waits to detect a slight variation on the monitor. The tools interpret the variation as the representation of the response to the CIA user's request. These external devices have a photodetector to measure variations in light intensity. These proposals place the photodetector on the user's monitor to make the measurements. Leo Bodnar Electronics offers devices that allow one to obtain interactivity time through emulated interactions, which are used by researchers to evaluate the QoE in cloud gaming. The device in [55] supports video signals up to full-HD quality, whereas the model in [56] allows for signal measurements at 4K resolution.

Johnsen [57] (UXMeter) and the NVIDIA Latency Display Analysis Tool (LDAT) [58] offer similar external devices based on light-intensity variations. They allow for the detection of user–mouse interactions. Their external devices connect to the user's PC via USB and receive the user's interactions by directly connecting the mouse to them via a USB port. The NVIDIA solution not only detects the interactivity of the display device but also provides a mini-jack audio input to evaluate the audio delay.

Ideally, in remote desktop or cloud gaming CIAs, the user's screen should be identical at the same instant of time as that of the cloud server. However, in practice, this is impossible due to the delay introduced by the network. Some authors evaluated the time difference between the screen representations of the cloud server and the client device. These QoE quantification proposals base their approaches on the fact that the shorter the time difference, the better the user experience. The cloud server of a remote desktop or cloud game sends screen updates to the client as fast as possible. The server must adjust its screen refresh rate based on the available resources (e.g., network bandwidth or CPU usage of the computers involved). To avoid overloading clients, some solutions only send screen updates once the client confirms receipt of the previous update. During this time gap, the screen of the cloud server may have changed. Instead of sending individual screen updates for each change, the cloud server unifies all changes into a single screen update. In this way, the CIA merges multiple screen changes occurring at the same pixel location into its most recent pixel value.

To quantify this phenomenon, there are three alternatives in the literature. In [39], Hsu et al. proposed a method for evaluating cloud gaming. In their study, the authors used external cameras pointed at two monitors: one connected to the thin client and the other to the cloud server. To correctly identify on-screen updates, the proposed methodology inserts a colour bar at the top of the screen that unambiguously encodes each frame. During post-processing of the videos, the authors extracted the time difference between instances when the screens represented the same update. In this case, the authors did not provide an order of magnitude of the time difference that would allow the user to be satisfied with the CIA, and they required the client and the cloud server to be close to each other during the test. Finally, Shu Shi et al. [43] proposed their distortion-over-latency (DOL) metric, seeking to combine interactivity time and rendering quality in cloud gaming CIAs. To obtain the DOL, the authors calculated the distortion between the screen update rendered by the thin client and the ideal screen update before being sent by the cloud server. The DOL is the product of the MSE of these two screen updates and the time elapsed until the thin client receives the distorted frame (synthetic interactivity time). Tools can easily calculate the DOL offline by saving all the screen updates from the client and the cloud server. However, performing this calculation in real time is complicated because the thin client does not have the original screen update to calculate the distortion. Shu Shi et al. proposed utilising a thin client and a cloud server to perform online calculations. The thin client sends the distorted screen update and the synthetic interactivity time to the server along with the original screen update to calculate the DOL. The authors did not provide a relationship between the DOL and QoE values but stated that they needed more subjective testing to better understand this relationship.

There are, therefore, some common weaknesses in the proposals based on screen updates. In general, they do not provide an order of magnitude for the obtained metrics to ensure user satisfaction with QoE. Some of the proposals do not allow real-time operation, and others do not take into account possible measurement errors or obstacles in the methodology used. However, this cannot be known in all the proposals we analysed because some do not provide sufficient technical details to replicate their operations. Additionally, some proposals are dependent on a specific system, application, or environment, as they rely on utilising equipment, applications, or drivers.

5.2. Strategies Based on the Use of Slow-Motion Benchmarking

Among the strategies available in the literature, one stands out: slow-motion benchmarking from Nieh et al. [49]. Slow-motion benchmarking is a methodology for obtaining interactivity time for remote desktop CIAs from network traffic. Slow-motion benchmarking estimates interactivity time from patterns of network packets exchanged by both ends. When a user interacts with an application via a desktop, the thin client sends a request to the cloud server. This request generates an increase in network traffic. When the application located on the cloud server produces a response, it sends it to the thin client, again generating a spike in network traffic. Slow-motion benchmarking monitors this network traffic to obtain the elapsed time between the two traffic surges. If slow-motion benchmarking captures the traffic near the thin client, that elapsed time approximates the time between when a user interacts via a keyboard or mouse with the thin client and when they perceive a response on the screen. However, to avoid problems in detecting increased network traffic, slow-motion benchmarking requires a single ongoing request to the cloud server. This methodology requires instrumenting the thin client to refrain from initiating a new user interaction until it receives a response to the previous request. Therefore, it is not suitable for measuring the QoE of a user in real time but only serves to characterise the scenario under controlled conditions.

The authors suggested two safeguards for implementing the methodology. The use of a remote desktop protocol generates approximately constant traffic due to the communication between the thin client and the cloud server. An increase in this traffic may confuse the base traffic of the protocol used with the traffic generated by user integration. An implementation of the methodology must take this fact into account and establish a threshold above which it considers the detection of an interaction. Furthermore, the authors stated that the interactivity time obtained by slow-motion benchmarking is an approximation. The methodology does not consider the time that the thin client employs to detect the user interaction or to prepare and send the interaction over the network, nor does it consider the time that the client needs to reflect the detected update in the network traffic on the screen.

Several proposals used slow-motion benchmarking as the core of their research. Among them is the work carried out by Nguyen et al. [51]. The authors proposed a tool (VDBench) to evaluate and compare different remote desktop solutions. VDBench is based on slow-motion benchmarking and offers the interactivity time obtained from synthetic user interactions (as it does not come from real-time users). In addition, it offers test automation to evaluate the scalability of virtual machines on the cloud server. VDBench collects metrics on bandwidth usage, network loss, CPU, and RAM.

Other authors chose to use slow-motion benchmarking to quantify the performance of different remote desktop solutions in their studies. Alali et al. [33] referred to VD-DUT as the synthetic interactivity time obtained by slow-motion benchmarking. They also supplemented it with other metrics to assess the visual and sound quality experienced by users. In their study, 115 participants used four applications (image viewing, Skype, 3D image viewing, and video playback) via Windows RDP. In their study, they modified network conditions to generate packet losses while users interacted following the principles set by slow-motion benchmarking for a single network interaction. They simultaneously correlated objective measurements of visual quality, sound quality, and VD-DUT with opinion values by questioning the users under various boundary conditions defined by the authors. The work of Alali et al. [33] used the MOS scale.

Therefore, there are some disadvantages common to proposals employing slow-motion benchmarking. This strategy offers merely an approximation of interactivity time since it derives it from network packets. Furthermore, it requires only an ongoing request from the client to correlate with the server's response packet. Therefore, it is not suitable for real-time QoE measurement but rather for characterising scenarios in controlled laboratory conditions.

5.3. Strategies Based on Audiovisual Degradation Measures

Strategies based on audiovisual degradation measures are those that seek to evaluate the deterioration in visual or sound quality experienced by applications or services when they are utilised through a CIA rather than locally. We explain below how researchers quantify this phenomenon.

The literature on the evaluation of CIAs addresses different metrics for quantifying audiovisual degradation. One of the simplest is the number of frames per second (FPS) the user receives from a CIA. Hsu et al. [39], Penaherrera-pulla et al. [50], and Liu et al. [35] used the FPS as a measure of QoE to evaluate cloud gaming services. If FPS decreases, the authors interpreted that QoE also decreases. In their study, Penaherrera-pulla et al. demonstrated that the lower the number of frames per second, the lower the game scores for the players of a cloud gaming service. However, the FPS measure quantifies only the amount of information the client receives, not its quality. This is why, during their experiment, they used software to record the screen of the client and the cloud server. They then compared, frame by frame, both video sources to extract a new distortion metric calculated as the average of the mean squared error (MSE) between the server and client screen pixels for each recorded frame. This proposal also did not provide a relationship between the FPS or the MSE of the FPS and the QoE.

In the original slow-motion benchmarking work, Nieh et al. [49] focused on providing an approximate measure of interactivity time in remote desktop environments. In addition, the authors offered a mathematical expression to quantify the visual quality experienced by the user when consuming video through a thin client. They proposed a term named visual quality (VQ), which takes as a reference the amount of traffic generated when using a slowed-down version of the applications. The cloud server plays a video stream slowly enough so that the thin client can fully process each frame before it receives the next frame. The methodology records the traffic for each frame of the video as a reference traffic load for seamless playback without degradation. Thus, the visual quality is the ratio between the traffic generated by the CIA at a normal FPS rate and its ideal reference obtained by the tool with a single frame on the network. Some of the proposals mentioned above used this expression in their research to complement the information provided by the VD-DUT metric, for instance, Alali et al. [33] and Nguyen et al. [51].

In [38], Song et al. used a similar approach for quantifying QoE on remote desktops based on video quality. They started with the assumption that unlimited network bandwidth guarantees the best display quality. Thus, visual quality is the ratio between the traffic that CIAs generate in a bandwidth-constrained environment (the real scenario) and the traffic they generate under unrestricted bandwidth conditions. Although these proposals seek to quantify the visual quality experienced by users of CIAs, they did not clarify what value of the VQ metric satisfies users' QoE or propose a methodology to map it to a proper QoE scale.

The peak signal-to-noise ratio (PSNR) [89] and structural similarity method (SSIM) [90] are other metrics that allow for the evaluation of the visual quality of CIAs. Authors using the PSNR or SSIM need to compare the quality of the video stream from the cloud server and that of the user's screen. A low PSNR or SSIM value indicates a large difference in quality between the two video sources. However, the PSNR and SSIM, unlike the MSE, not only measure the frame-by-frame quality of the video sources but also reflect the desynchronization between them. Thus, a delay in receiving frames from the cloud server to the client results in a reduction in the PSNR or SSIM. For the PSNR, researchers must calculate the logarithm of the MSE for each pixel in the image. However, the SSIM is calculated using luminance, contrast, and image structure. The PSNR is more common because of its simple calculation and because it is a metric traditionally used in the video field. However, the SSIM has higher sensitivity compared to the PSNR for detecting small variations in video sources [91]. Magaña et al. [42] used the PSNR as a measure of QoE in remote desktop CIAs by comparing two video streams. They compared the Amazon Workspaces stream measured at the server with the received stream measured at the user's end. They then obtained the PSNR measurements to assess the impact of different network conditions. Magaña et al. mapped these PSNR measurements to an MOS scale value. In their study, Hsu et al. [39] employed the SSIM, PSNR, and FPS together to evaluate the QoE in cloud gaming services. Such proposals offer a promising way to assess QoE. However, they require development and optimisation to be able to compute the PSNR in real time.

The ITU-T proposed an algorithm to evaluate the quality of audiovisual content, called the perceptual evaluation of video quality (PEVQ), which some authors used to quantify QoE in CIAs [92]. The PEVQ is an algorithm designed for the evaluation of streaming video and is a solution that is not often adopted in the field of CIA research. The PEVQ is a standard metric that estimates video quality by providing an MOS value at the output. This requires comparing the original, undegraded reference signal with the signal received by the client. Proposals using this metric require instrumentation from both video sources to calculate it. The ITU-T offers the PEVQ not only for the assessment of visual degradation but also for audio degradation. In [93], the perceptual objective listening quality analysis (POLQA) was used to measure the audio degradation in content after receipt by the user. The algorithm uses the audio source and received audio source for its calculation to evaluate the sampling frequency, compression, and synchronisation. The metric provides an output with an estimated value of the MOS experienced by the user. In [37], Dong et al. employed the PSNR, FPSs, and SSIM to analyse the user experience in a remote residential desktop environment. In [46], Wang et al. also employed the SSIM, PSNR, and PEVQ in their study on cloud gaming. The authors proposed a strategy to optimise game rendering by delegating part of it to the thin client. Their proposal utilised several configurable parameters. The study analysed the impact of the values of these parameters and the effectiveness of their proposal compared to the state of the art. using audiovisual degradation metrics. These metrics require the video sources of the cloud server and the client, so their implementation in real environments is difficult. However, they provide a concrete QoE value.

Regarding audio, in [33], Al Ali et al. used three different metrics to compare original and user-perceived audio: Weighted Spectral Slope (WSS) [94], Log-Likelihood Ratio (LLR) [95], and Virtual Speech Quality Objective Listener (ViSQOL) [96]. The study focused on remote desktop environments. The WSS provides a distance measurement between the original signal and the one received by the user by obtaining a weighted difference between spectral slopes in different frequency bands. The spectral slope is obtained as the difference between adjacent spectral magnitudes in decibels. The LLR also provides a measure that quantifies the spectral differences between the reference signal and the compressed signal. The ViSQOL metric models the perception of human speech quality using a spectrotemporal measure of the similarity between a reference and a test speech signal. Similar to previous proposals, these metrics require the audio sources of the cloud server and the client, which makes their implementation in real environments difficult.

Therefore, there are some common drawbacks of the proposals based on audiovisual degradation measures. The main one is that the proposed measures require obtaining audio or video signals from both the client and the server to compare degradation. This condition can be complex and hinders real-time QoE assessment. However, these proposals commonly provide a specific QoE value, which is not very common in other types of strategies.

5.4. Strategies Based on the Instrumentation of Programming Code

This strategy refers to proposals that require modifying the source code of applications or services running on at least one element of the CIA infrastructure to provide QoE measures.

A number of authors tried to evaluate the interactivity time or responsiveness of CIAs through CIA instrumentation. Unlike strategies based on screen updates, authors who opted for code instrumentation did not need to analyse the graphical content of the screen to determine when the response to a user request was represented. Instead, they directly instrumented the CIA code to add timestamps when functions of interest were executed. In this case, the authors aimed to identify when the server received a user interaction and when the client received the graphical response sent by the cloud server.

In [35], Liu et al. instrumented code to evaluate QoE in cloud gaming CIAs. They proposed a tool called Pictor for benchmarking cloud gaming services. Pictor has two components. The first one aims to automate and replicate the comparative tests. To do this, Pictor generates interactions with the CIA as a real user would. The second component is responsible for evaluating the behaviour of the service in the face of these emulated interactions. The first component uses neural networks for its operation, so it needs to be trained on labelled recordings from cloud gaming services. Accordingly, the researchers instrumented the cloud gaming CIA client TurboVNC [97] to capture user interactions. Additionally, Pictor uses image-processing techniques to identify the different graphical objects present in the service with which the user can interact. The researcher must label each of these objects during the training phase. The collected information makes up the training set for a recursive neural network, specifically, long short-term memory (LSTM). As a result of training, the neural network can interact with the cloud gaming service in a real scenario. After automating and replicating the interactions of the CIA, the authors subjected the service to different boundary conditions to evaluate its performance. The second component of Pictor allows for comparing configurations and services with each other. Due to the instrumentation, Pictor can obtain the interactivity time. Specifically, the authors instrumented the service to monitor calls to several common graphics library functions (OpenGL [98]), allowing them to determine how long the CIA takes to render graphics objects after an interaction.

Jahromi et al. [36] sought to measure interactivity time by instrumenting the HTML and JavaScript code of interactive web applications. When the user interacts with an interactive web application, it triggers the rendering of various HTML components. This approach requires a taxonomy of all HTML elements on the web. In a previous work, the researchers determined how many elements rendered after a given interaction represented the final graphical state of the associated response. For each possible interaction with the CIA, the authors calculated the thresholds for the triggered events. Using this instrumentation, the authors monitored user input and the number of HTML elements rendered. When the CIA receives an interaction, the instrumented code calculates the moment at which the interactive web application renders all HTML elements of the associated response. The output time metric is an approximation of the interactivity time, as it does not account for the time required by the browser to transfer the screen refresh to the operating system.

In the field of interactive web applications, the literature presents various alternative time metrics to interactivity time achieved through the instrumentation of their code to represent QoE. Saverimoutou et al. [40] presented a tool called Web View, which automates the collection of several of these metrics. The time to first paint (TTFP) is the time it takes for the first pixel to appear on the user's browser screen after the first interaction with the web service. To calculate the TTFP, the researchers instrumented a CIA to obtain the time elapsed between the establishment of the first connection at the network level and the instant the first HTML element starts to be rendered. The TTFP is a metric the authors associated with QoE. Even if it does not measure the time to render the final response, it captures a scenario where the user becomes impatient, thinking that the CIA is stuck or not working properly if they do not see any kind of update on the screen [40].

The Page Load Time (PLT) represents the duration between the start of browsing and the entire web page loading. It is calculated as the time between the establishment of the first connection at the network level and when the page finishes loading and renders the last HTML element. The PLT was proposed by Saverimoutou et al. as a measure of QoE. However, some HTML elements of the web page are not visible unless one scrolls to the non-visible part of the CIA. This is why Web View obtains the Time for Full Visible Rendering (TFVR) metric as an alternative. The TFVR represents the time it takes for the interactive web application to render the part directly visible to the user without scrolling. The instrumentation code automates the process of determining when each of the elements is rendered and their respective positions on the final web page. Some HTML elements, such as images, may be cached, and the instrumentation takes this phenomenon into account. Once the coordinates and dimensions occupied by each element of the DOM have been determined, the tool decides which elements and at what time the browser renders them in the visible part.

Additionally, in the field of interactive web applications, there is the work by Hossfeld et al. in [44], which is similar. However, the work focuses on the speed index (SI) metric. This is a metric that evaluates the time it takes to render a change on the screen in web applications, but may also be applicable to CIAs. Specifically, the SI quantifies the speed with which screen updates are rendered. To do this, by instrumenting the CIA code in the client, it measures the duration between the rendering of two known graphical states characterised by the histogram of their pixels. Although the SI is not an approximate or proportional measure of interactivity time, it is sometimes used as a representative metric of QoE. For this purpose, Hossfeld et al. proposed a mathematical model to map different SI values to an MOS scale.

Therefore, there are some common drawbacks of proposals based on the instrumentation of programming code. Primarily, it can be challenging to instrument applications, operating systems, or drivers. This implies that some of the proposals are less generalizable and, consequently, in certain cases, the measurements they provide are approximations.

5.5. Strategies Based on Indirect Measures

This strategy refers to proposals that quantify QoE through indirect measures. Indirect measures are those obtained from other intermediate measures that, in principle, are not directly related to QoE. We explain below the most common intermediate measures used by researchers to assess QoE.

In [47], Wehner et al. proposed an alternative to avoid the instrumentation of interactive web applications and obtain the SI without incurring high computational costs. Although the SI traditionally requires the instrumentation of the CIA, the authors suggested obtaining the metric through indirect measurements. Their proposal used network traffic and artificial intelligence models. To train the models, Wehner et al. extracted the number of bytes, number of packets, arrival times, and bandwidth generated by the CIA. From these values, they calculated statistics such as the mean, maximum value, and sum. In the study, they tested different models: k-nearest neighbours (KNN), decision trees, random forests, extreme gradient descent (XGB), LSTM neural networks, and gated recurrent units (GRUs). The LSTM networks provided the best results for estimating the SI from network traffic in interactive web applications.

In [48], Graff et al. defined the relationship between different main key performance indicators (KPIs) of the indirect measures that are relevant to determining the QoE in cloud gaming CIAs. However, the QoE model resulting from the experiments was a conceptualisation and did not provide a final definition specifying which KPI transformations are necessary to obtain a real QoE value.

Li et al. [41] proposed latency as an indirect measure. The authors aimed to evaluate the QoE in remote desktop CIAs by employing active traffic measurements. The study suggested using the response time of ICMP (ping) requests from the client to the cloud server. The approach requires creating a baseline by pinging a cloud server. Following this previous work, Li et al. used this baseline to evaluate the QoE in CIAs in real scenarios. When the server is more loaded or network administrators add new virtual machines to the server, some ping requests will show higher times. Their indirect latency measure assesses these anomalous high values that deviate from the norm. However, again, the paper did not clarify the relationship between the indirect measure and QoE. Furthermore, it did not use real users to evaluate whether there was a real relationship between the peak latency phenomenon and a decrease in QoE.

Some proposals offer proprietary measures of QoS as proxy measures for QoE quantification. The most common ones are CPU, GPU, RAM, disk, and network usage. These metrics are traditionally used to evaluate equipment performance but are not necessarily directly related to the user experience. Some authors obtained these metrics from the client or the cloud server. In [31], Casas et al. used network metrics to evaluate QoE in remote desktop and interactive web page CIAs. In their study, the authors analysed the QoE in different commercial CIAs. They subjected the service to different boundary conditions and obtained network metrics such as upstream bandwidth, downstream bandwidth, and RTT. The authors attempted to map these metrics to QoE using laboratory tests with real users. In these tests, the researchers also calculated the synthetic interactivity time, which can be derived from network measurements, such as the RTT. However, the study did not provide much detail about how to obtain the synthetic interactivity time or its measurement error.

If CPU, GPU, RAM, or disk usage is high, the different tasks running on a computer may be queued, and the user may perceive a slowdown in the CIA services consumed. When there are no processes running on the computers, the use of computing resources remains stable. A significant variation in the stability of the usage of these measures may be indicative of the start of new processes on the computers in the CIA architecture. Liu et al. [35] measured CPU and GPU usage in their study of cloud gaming. Laghari et al. designed and developed [53] a platform to evaluate QoE in any CIA. The objective was to check whether the parameters offered by the service were consistent with the service level agreement (SLA). The values compared were the CPU, RAM, and GPU usage of the client, as well as the bit rate of the connection between the client and the CIA cloud server.

The problem with these proposals is that none of the authors established which values of the measurements they obtained are sufficient to guarantee QoE. Although there was a relationship identified between QoS and QoE, these studies did not use real users subjected to different boundary conditions to verify and evaluate the correlation between QoS and QoE. Mahmud et al. [54] developed a model to guarantee QoE in any type of CIA. The authors argued that by optimising different metrics, the CIA user would have a good QoE. Their model is based on fuzzy logic and optimises two indirect measures called the Rating of Expectation (RoE) and Capacity Class Score (CCS). Using the values of these metrics, the authors determined the ideal location for the CIA's cloud server to guarantee an accurate QoE result. The RoE categorises the CIA's requirements based on its access rate (slow, normal, fast), expressed in user accesses per second; the computational resources it needs (small, regular, large), expressed in CPU cores; and the processing time required (stringent, moderate, flexible), expressed in milliseconds. The CCS, on the other hand, reflects the state of the cloud server, categorising the cloud server requirements by the RTT (short, typical, lengthy), expressed in milliseconds; the availability of computational resources (poor, standard, rich), expressed in CPU cores; and the processing speed (least, average, intense), expressed in thousands of instructions per second (TIPS). The Mahmud et al. model allows for selecting the best location for the cloud server and for understanding in real time whether the QoE requirements of a particular CIA can be met by thresholding the RoE and CCS metrics. However, the proposal did not study the relationship between the selected metrics used to calculate the RoE and CCS and the QoE perceived by users.

Therefore, proposals based on indirect measures have some drawbacks. Some studies lack details on how to reproduce their QoE assessment methods. Sometimes, the relationship between QoE and the proposed measures is not specified. In addition, there is a risk of confusing QoS measures with QoE in certain cases.

5.6. Comparison of Strategies

Table 4 provides a summary of the strategies for QoE measurement in CIAs. It is worth noting that the human perceptions related to QoE are considered in the proposals discussed within each strategy. Among these, only the proposals based on indirect measures attempt to cover the three human perceptions that influence QoE. This is because this strategy encompasses a variety of proposals. However, it is concerning that no strategy unanimously offers a real measure of QoE. Only isolated proposals make an effort to establish a mapping between their proposed metrics and a QoE scale such as the MOS. Moreover, among the proposals that consider the human perceptions, few specify thresholds for the provided metrics. Only a small number of proposals belonging to the screen-based approach do so. This lack of specification makes it challenging for administrators to implement the methodologies suggested by the proposals. Furthermore, both the slow-motion benchmarking strategy and the indirect methods strategy involve approximations for the measures they propose. In the case of slow-motion benchmarking, an incomplete interactivity time is obtained, referred to as synthetic interactivity time. In the case of the strategy based on indirect methods, the proposals involve inaccurate mappings between QoS parameters. Finally, Table 4 lists some of the advantages and disadvantages of the proposals described in the text.

Table 4. A summary of strategies for QoE measurement in CIAs and systematic discussions.

Strategy Based on	Human Perceptions Considered			Real QoE Metric	Metric Thresholds	Strategy Accuracy	Common Proposals: Advantages	Common Proposals: Drawbacks
	Visual Quality	Audio Quality	Interactivity					
Screen updates	×	×	✓	None	Rarely	High	· Real-time operation · Widely accepted	· May require external devices · Lack of mapping between interactivity and QoE
Slow-motion benchmarking	✓	×	✓	Mostly none	None	Estimation	· Widely accepted	· Required modification of CIA user behaviour · Target metric approximation · No real-time operation
Audiovisual degradation measures	✓	✓	✓	Mostly none	None	High	· ITU-T standardised proposals	· Need both audio/video sources from the cloud server and client
Instrumentation of programming code	×	×	✓	Mostly none	None	High	· Many output metrics · Quantify subtle aspects of interactivity	· Lack of mapping between interactivity and QoE · Proposals difficult to generalise to other CIAs · Requires access and knowledge of source code
Indirect measures	✓	✓	✓	Mostly none	None	Estimation	· Easily obtainable input metrics	· Misuse of QoS metrics · Lack of real users · Target metric approximation

6. Stages of the Quantification of QoE in CIAs

In this section, we differentiate between the three main stages that all strategies must address in QoE assessment: collection of metrics, processing, and generation of the output. Figure 5 represents the sequential process of these three stages.

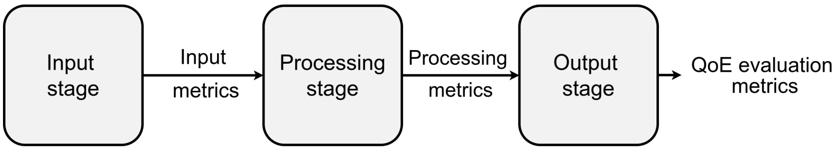

Figure 5. Sequential process of QoE evaluation.

In the input stage, the proposals obtain metrics for some of the infrastructure components, as depicted in Figure 4. Subsequently, the proposals process this input information in the second stage. Finally, in the output stage, new metrics are generated, more or less related to the QoE of the CIA being evaluated. We group the proposals according to the metrics used by the authors in each of the stages.

6.1. Input Stage

We present a classification of the information extracted from the proposals in the literature with respect to the sources of information or starting metrics extracted from the elements of the CIA's infrastructure. In particular, some researchers aimed to extract information related to the human perceptions discussed in Section 2. In some cases, these were the inputs used directly in the next stage of processing, whereas in others, the tools performed transformations of the metrics to obtain other input metrics. In Figure 6, the 21 metrics or sources of input information collected from the state of the art are grouped into four main categories.

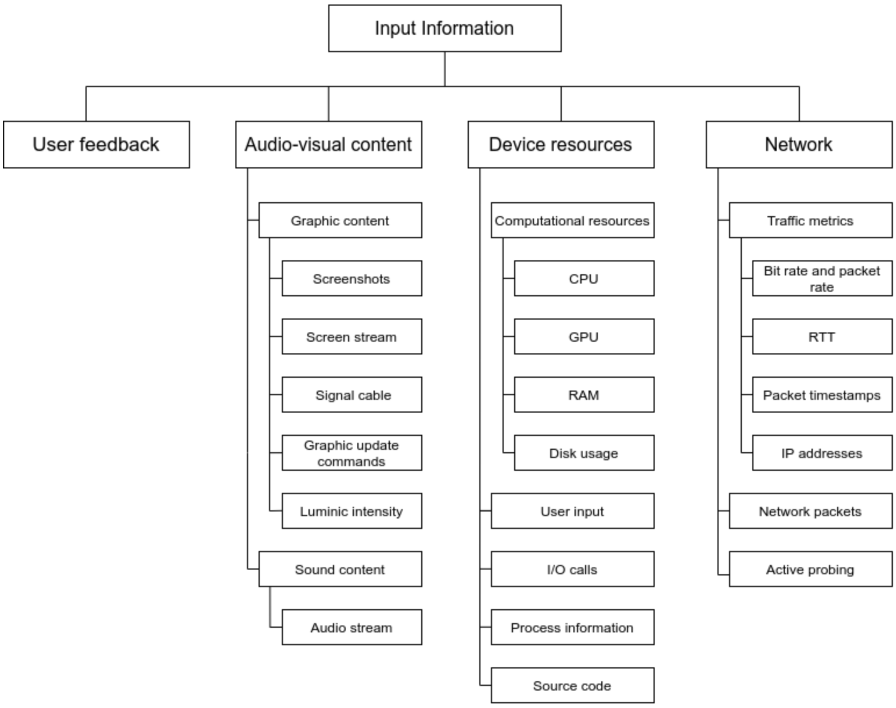

Figure 6. Grouping of input stage metrics into four types of information sources.

The four main sources of information are as follows:

- Audiovisual content: The tools extract the input information about the graphic or audio content from the CIA the user is consuming.

- Device resources: The input information comes from the computational resources of the elements of the CIA architecture.
- Network information: Incoming information comes from network traffic.
- User feedback: Tools extract input information directly from user impressions.

Table 5 summarises the proposals we discussed in Section 5 and identifies the different types of input sources used, along with the specific metrics. Each proposal may use several sources of input information simultaneously. The table categorises the proposals into two broad types: specific or general scopes. This categorisation distinguishes works that are generalizable to any type of CIA (general scope) from those in which the authors specialise in quantifying QoE in specific CIAs (specific scope). In the case of remote desktops, proposals that use input metrics independent of the remote desktop protocol used by the CIA are part of the general scope group. Conversely, proposals that dissect any given remote desktop protocol to obtain the input metrics are part of the specific application scope.

Audiovisual content allows researchers to obtain sources of information and metrics related to the visual and audio perceptions of the CIA user. To obtain metrics or sources of information on visual perception, the source can be the graphic content represented by the CIA. Some proposals attempted to obtain measurements in a timely manner, opting to obtain screenshots of specific elements of the architecture as a source of information. Other proposals chose to obtain measurements continuously, using screen streams as a source of information. To obtain screenshots or screen streams, researchers either instrumented the architecture or externally recorded some display elements. When proposals obtained screenshots or screen streams on the client side, they obtained metrics related to the user's final receipt. However, proposals that obtained their source information from the cloud server extracted information prior to any visual degradation generated by the infrastructure. As discussed in Section 5, some researchers used graphical content to obtain metrics such as visual quality or frames per second (FPS). These tools can obtain these metrics from graphical content by passively monitoring the connections between graphical representation elements (e.g., a screen monitor). Strategies that instrument the wiring require additional custom hardware.

In remote desktop or cloud gaming CIAs, the cloud server can send compressed or uncompressed graphical content over the network. Additionally, the cloud server may be able to communicate with the client by employing graphical primitives of the operating system. In such a case, the cloud server indicates how and which regions of the client's screen it will update. Graphical primitives save bandwidth but require an understanding of the client's graphical libraries and the cloud server. Even when the graphical content was not accessible in real time, some proposals used graphical update primitives. These proposals measured the frequency of graphical updates, either when sent by the server or received by the client, as well as the percentage of the user's screen updated by each primitive. Other proposals opted for light-intensity measurements. This source of information was interesting in cases where researchers could not instrument the client, network, or cloud server. Some authors used comparative light-intensity measurements to detect the frequency with which changes occurred on the screen.

To obtain metrics or information sources from sound perception, the information sources are the sound content reproduced by the CIA. When the proposals obtained the audio from the client, they obtained information related to the user's final reception. However, proposals that obtained audio from the cloud server extracted information prior to any sound degradation generated by the infrastructure. The authors sometimes chose to use both sources of audio information in combination.

Table 5. Literature proposals and the input metrics used.

Ref.	User Feedback	Audiovisual Content						Device Resources								Network					
		Graphic Content					Sound Content														
								Computational Resources								Traffic Metrics					
		Screen Stream	Screenshots	Graphic Update Commands	Luminic Intensity	Signal Cable	Audio Stream	CPU	GPU	RAM	Disk Usage	User Input	I/O Calls	Process Information	Source Code	Bit Rate and Packet Rate	RTT	Packet Timestamps	IP Addresses	Raw Packets	Active Probing
General scope application																					
Casas et al. [31]	Yes		Yes									Yes				Yes	Yes				
Wang et al. (2017) [46]			Yes																		
Laghari et al. (2018) [53]	Yes	Yes						Yes		Yes				Yes		Yes			Yes		
Magana (2019) [42]			Yes																		
Johnsen (2019) [57]					Yes							Yes									
Mahmud et al. (2019) [54]								Yes	Yes	Yes	Yes			Yes			Yes				
Li et al. (2019) [41]																					Yes
Nvidia (2020) [58]					Yes							Yes									
Wehner et al. (2020) [47]					Yes	Yes															
Leo Bodnar Electronicas (2021) [55]					Yes	Yes															
Leo Bodnar Electronicas (2021) [56]					Yes	Yes															
Hsu et al. (2021) [39]		Yes						Yes								Yes					
Penaherrera-pulla et al. (2021) [50]			Yes																		
Graff et al. (2021) [48]																Yes	Yes	Yes			
Arellano-Uson et al. (2021) [9]			Yes									Yes									
Dong et al. in [37]		Yes					Yes									Yes		Yes		Yes	
Specific scope application																					
Nieh et al. (2003) [49]		Yes														Yes		Yes			
Shushi et al. (2011) [43]			Yes									Yes						Yes			
Nguyen et al. (2015) [51]		Yes														Yes		Yes			
Varghese et al. (2018) [52]			Yes									Yes									
Song et al. (2018) [38]								Yes								Yes		Yes			
Hossfeld et al. (2018) [44]															Yes						
Saverimoutou et al. (2019) [40]															Yes						
Alali et al. (2019) [33]	Yes						Yes									Yes		Yes		Yes	
Jahromi et al. (2020) [36]															Yes						
Liu et al. (2020) [35]		Yes						Yes	Yes			Yes	Yes			Yes		Yes			
Kumar et al. (2021) [34]			Yes									Yes									
Exoprise (2024) [45]			Yes																		

For device resource information, Table 5 groups the eight related input metrics. Some researchers derive metrics from the source information of the computational resources of the architecture elements. The most common metrics are CPU, GPU, RAM, and disk usage. Another source of information is user input. Some researchers obtain metrics related to when the user interacts with their devices via a keyboard or mouse. User input is one source of information that provides metrics on when they occur, how often they occur, the type of action performed (e.g., a mouse click or a key press), and the locations of the interactions (mouse coordinates). Another source of information is the system calls that applications use to communicate with the operating system. Some approaches choose to monitor the input and output (I/O) of these calls to evaluate the content, their frequency, or the time elapsed between the system receiving the input and generating the output. Some researchers use not only these times as a source of information but also the behaviour of the executed processes as input metrics. This source of information is different from I/O calls, as it allows the tools to obtain metrics of applications: the physical resources it uses, the libraries it calls during its operation, and the execution times of its internal processes. Some authors extract the I/O and process information from both the client and cloud server through the APIs of their OS. Another source of information is the source code of the applications that the literature uses to obtain metrics related to their behaviour. Using the source code, some proposals calculate, among other things, metrics such as the rendering time of an element of a web application, the time at which a certain response is sent or received by the client, or when a screen update order is sent or received. This source of information requires instrumenting the applications (APM).

The network allows input metrics to be obtained from the traffic generated by the CIA. Some approaches monitored traffic to obtain network traffic metrics, such as bit rate, packet rate, timestamps, or IP addresses present in the traffic. Other proposals used the raw content of network packets. This source of information requires knowledge of the communication protocol in use to extract parameters from the messages exchanged between the client and the cloud server. This enables the determination of the specific packet that contains a screen update, a user action, or the compression codec used in the CIA. Currently, it is challenging to use this source of information, as most network traffic is encrypted. Other proposals used active polling by generating synthetic network traffic directed at infrastructure elements and evaluating the responses of the target devices. Active polling allows for deriving metrics, such as response times or the status of the client or cloud server, from response codes.

User feedback is a direct source of input information related to QoE in CIAs. Some proposals in the literature collected user feedback through forms or surveys, allowing users to express their opinions when the study exposed them to different boundary conditions. Users rated their experience on a predefined scale or in their own words. The most frequent scales used were the ACR or MOS, as mentioned above. Other scales include the Standard Deviation of Opinion Scores (SOS) [99] and the Net Promoter Score (NPS) [100]. The SOS is a measure that states that in properly conducted QoE evaluation tests, user ratings differ very little, as all users experience the same boundary conditions. The NPS, on the other hand, assesses not only user satisfaction but also user loyalty. The NPS determines how likely a person is to recommend a brand, company, product, or service to another person.

6.2. Processing Stage: Computing QoE

Here, we present a classification of how the proposals in the literature process the input stage metrics:

- Ad hoc heuristics: Processing techniques created to obtain an output metric that is not generalizable to objectives other than the quantification of QoE in CIAs.
- Image processing: A set of techniques applied to digital images to transform them or extract features.
- Artificial intelligence: Processing techniques that use a set of training data to automatically build a model.

Table 6 summarises the proposals presented in Section 5 and groups them by processing technique. Some proposals use multiple processing techniques simultaneously.

Table 6. Proposals from the literature and the processing techniques used.

Ref.	Ad Hoc Heuristics	Image Processing	Artificial Intelligence
Nieh et al. (2003) [49]	Yes		
Shushi et al. (2011) [43]	Yes		
Casas et al. (2014) [31]	Yes	Yes	
Nguyen et al. (2015) [51]	Yes		
Wang et al. (2017) [46]		Yes	
Varghese et al. (2018) [52]		Yes	
Song et al. (2018) [38]	Yes		
Hossfeld et al. (2018) [44]	Yes		
Laghari et al. (2018) [53]	Yes		
Magaña et al. (2019) [42]		Yes	
Saverimoutou et al. (2019) [40]	Yes		
Johnsen (2019) [57]		Yes	
Alali et al. (2019) [33]	Yes		
Mahmud et al. (2019) [54]	Yes		
Li et al. (2019) [41]	Yes		
Jahromi et al. (2020) [36]	Yes		
Nvidia (2020) [58]	Yes		
Liu et al. (2020) [35]	Yes	Yes	Yes
Wehner et al. (2020) [47]			Yes
Leo Bodnar Electronics (2021) [55]	Yes		
Leo Bodnar Electronics (2021) [56]	Yes		
Hsu et al. (2021) [39]		Yes	
Penaherrera-pulla et al. (2021) [50]		Yes	
Graff et al. (2021) [48]	Yes		
Kumar et al. (2021) [34]		Yes	
Arellano-Uson et al. (2021) [9]	Yes		
Dong et al. in [37]	Yes		
Exoprise (2024) [45]		Yes	

Processing techniques based on ad hoc heuristics varied. Some proposals calculated ratios from the input metrics, related several of them through regression techniques or mathematical adjustments, or directly thresholded one or more of the input metrics based on studies that related their thresholds to QoE degradation. On the other hand, other works used the metric obtained through processing as the representative value of the QoE assessment. In these techniques, researchers did not determine whether a processing metric was sufficient to guarantee QoE. These works were based on the hypothesis that optimising the obtained metrics (maximising or minimising, as the case may be) is a measure for guaranteeing sufficient QoE to users.

Processing techniques based on image processing used digital images or image sequences as inputs. These processing techniques also varied. Some proposals involved comparing images to determine if they were the same, calculating degradations, or identifying modifications in different elements of a CIA's architecture. Authors achieve this by comparing the bitmaps of an image pixel by pixel or using discriminators to check whether they match previously stored reference images. Some proposals calculated hashes (MD5, CRC, SHA-1, etc.) for discriminators, while others modified the processed images to insert an encoding that allows for unequivocal identification of each image or frame, differentiating one from the others with unique identifiers. These identifiers enable the tools to determine whether two images are the same without the need to check all their pixels. However, pixel-by-pixel comparison allows for the calculation of metrics such as which regions differ or the percentage of updated pixels. To calculate the image degradation experienced by users on the client side, traditional image-processing techniques offer comparative measurements such as the MSE, PSNR, or SSIM. In some cases, image processing is used to identify specific elements within them. To achieve this, strategies must employ computer vision to recognise objects or use character recognition such as OCR to identify text.

Artificial intelligence-based processing techniques employed machine learning (ML) or deep learning (DL) techniques as the core of their input parameter processing. These algorithms differ from traditional combinational methods because they employ a much larger number of input parameters. These approaches attempted to find complex relationships among input parameters and automatically relate them to QoE. To do this, researchers generated large datasets that they used to train the models. Among the algorithms we analysed are support vector machines, random forests, naive Bayes, decision trees, k-nearest neighbours, extreme gradient descent, and neural networks such as long short-term memory, recursive, or gated recurrent units.

6.3. Output Stage

Below, we present a classification of the proposals for evaluating QoE in CIAs based on the metrics they contribute to the output stage. We group the proposals into five broad groups. The aim of this section is to discern which research directly assesses the human perceptions outlined in Section 2 and actually provides QoE-related metrics. Table 7 summarises the proposals we discussed in Section 5 and groups them by their output metrics, where more than one can be used simultaneously. In this section, we briefly review these metrics:

- Video metrics: Metrics that quantify the human perception of visual quality.
- Audio metrics: Metrics that quantify the human perception of sound quality.
- Time metrics: Metrics that quantify the human perception of interactivity or responsiveness.
- QoE metrics: Metrics that directly summarise the user experience.
- Other QoS metrics: Quality of service metrics that do not relate to human perceptions or QoE.

Video metrics assess the quality of the graphical content consumed by the CIA user. It was common to provide measures related to the number of frames per second perceived by the user, the number of frames lost or skipped by the architecture following its delivery on the cloud server, or the percentage change of a frame from one screen update to the next. Other researchers proposed output metrics directly related to their processing stage. Some output metrics were proprietary to each work, whereas others were well known (e.g., PSNR or SSIM) or endorsed by institutions such as ITU-T (e.g., PEVQ).

Audio metrics evaluate the quality of the audio content consumed by the CIA user. In the output stage, some proposals utilised well-known sound-quality metrics such as WSS, LLR, or ViSQOL, as discussed above. As in the case of video metrics, some authors also used ITU-T-endorsed metrics such as POLQA.

Among the proposals that utilised output time metrics, a large number of researchers tried to extract interactivity time: the elapsed time perceived by the user from the moment they interact with the CIA until the representation of the associated response. Meanwhile, other works used approximations (synthetic interactivity time). Some proposals also provided a measure of the time it takes for users or CIA applications to complete a sequence of tasks. Others opted to provide visual load times, often for interactive web applications (e.g., speed index, first paint, page load time, or time for full visible rendering). These proposals did not directly measure interactivity time but indirectly measured the human perception of responsiveness.

Table 7. Proposals from the literature and the output metrics provided.

Ref.	Video Metric	Audio Metric	Time Metric	Other QoS Metrics	QoE
Nieh et al. (2003) [49]	Yes		Yes		
Shushi et al. (2011) [43]	Yes		Yes		
Casas et al. (2014) [31]			Yes	Yes	
Nguyen et al. (2015) [51]	Yes		Yes		
Wang et al. (2017) [46]	Yes				
Varghese et al. (2018) [52]			Yes		
Song et al. (2018) [38]	Yes		Yes		
Hossfeld et al. (2018) [44]			Yes		Yes
Laghari et al. (2018) [53]				Yes	Yes
Magaña et al. (2019) [42]	Yes				Yes
Saverimoutou et al. (2019) [40]			Yes		
Johnsen (2019) [57]			Yes		
Alali et al. (2019) [33]	Yes	Yes	Yes		Yes
Mahmud et al. (2019) [54]				Yes	
Li et al. (2019) [41]			Yes		
Jahromi et al. (2020) [36]			Yes		Yes
Nvidia (2020) [58]		Yes	Yes		
Liu et al. (2020) [35]				Yes	
Wehner et al. (2020) [47]			Yes		
Hsu et al. (2021) [39]	Yes		Yes	Yes	
Leo Bodnar Electronics (2021) [55]			Yes		
Leo Bodnar Electronics (2021) [56]			Yes		
Penaherrera-pulla et al. (2021) [50]	Yes				
Graff et al. (2021) [48]			Yes	Yes	
Kumar et al. (2021) [34]			Yes	Yes	
Arellano-Uson et al. (2021) [9]			Yes		
Song et al. (2018) [38]	Yes	Yes			Yes
Exoprise (2024) [45]			Yes		

Through ACR/MOS scale values, QoE metrics often provided a real quantification of user satisfaction. These proposals obtained these values directly by asking CIA users or using proprietary or previous models to map metrics from the other categories of output metrics onto the ACR/MOS scales.

The most commonly used QoS metrics at the output stage were those indicative of the state of computational resources such as CPU, RAM, GPU, or disk usage. Other proposals provided network metrics such as bandwidth usage, bytes sent or received, number of packets used, or RTT. The main deficiency of these output metrics is that most works did not establish the relationship between QoS and QoE metrics.

7. Open Issues and Lessons Learned

Cloud-based interactive applications and services are becoming increasingly prevalent. The pandemic situation caused by COVID-19 has strongly accelerated the deployment of some of these solutions, particularly remote desktop services. The general public is increasingly using solutions with high interactivity requirements. This trend must be accompanied by efforts to assess the QoE that CIAs offer. Service administrators need to ensure good QoE so that user productivity is sustained. We observed that over the last three years, the number of research studies addressing the evaluation of QoE in CIAs has increased. We hope that given the current situation, further progress will be made in this field.

Presently, researchers need to collect input metrics for their proposals from CIA infrastructure elements. There is a need for CIA developers to provide specific APIs and software for assessing the performance of CIAs deployed in public or private cloud environments. This would streamline the work of service administrators and allow them to adjust the computational resources of the CIA architecture in real time. In addition, communication protocols are, in most cases, proprietary. This makes it difficult to develop methodologies that can be generalised to more than one CIA. In view of this fact, methodologies such as slow-motion benchmarking require instrumenting clients and modifying user behaviour to extract QoE measurements. Therefore, the measurements may not be representative. In addition, many of the proposals we evaluated have a particularised approach for specific CIAs. Hence, it is necessary to develop new and more general strategies.

A number of proposals bundle QoS metrics and offer them as a new measure of QoE quantification. Authors such as Jahromi et al. [36] and Casas et al. [31] studied the correlation between their input metrics and the user's final perception. To take the step of relating QoS to QoE, the studies conducted laboratory tests with a large number of users who experienced different boundary conditions while answering questions about their usage experience. However, a large number of the proposals we analysed did not demonstrate any direct relationship between their QoS metrics and the QoE evaluation of the CIA. This is the main shortcoming we identified. This missing step is costly and time-consuming due to the need for laboratory tests, but it will enable one to obtain truly representative QoE metrics.

To overcome this deficiency, it is necessary to conduct controlled lab experiments [101]. The procedure has been standardised and protocolised by ITU-T [102–104]. According to these standards, it is necessary to control the conditions of the overall evaluation process. The experimenter must control the content and context of the process and inform and observe the user at any moment during the lab tests. However, other possibilities are becoming more and more common when it comes to assessing the performance of networks and services from a QoE end-user perspective. Some authors understand that the user experience is also influenced by factors such as the context of use, the preferences of some services over others, or the device a user is used to [85]. This is why another possibility is to conduct controlled experiments in the user's own environment. These experiments would yield more realistic results that complement those carried out in the laboratory [105], even though they complicate the procedure by having to be conducted in the user's environment.

Most proposals lack an assessment of the performance of the measurement system, including its impact on the systems it operates on. Only a few studies propose methodologies, such as instrumenting elements of the CIA architecture with reduced computational resources (e.g., thin clients), but it is not common for researchers to thoroughly evaluate the potential impact on CPU, RAM, or disk usage. Furthermore, many proposals are proof-of-concept developments and often lack detailed implementation descriptions.

Finally, the proposals are often difficult to compare. In many cases, studies do not provide information about measurement errors. Therefore, there is a need for well-defined methodologies to ensure the comparability and reproducibility of new proposals.

The analysis of state-of-the-art papers enables us to derive several crucial lessons to be considered in the design of future proposals. It is imperative that the measurement of QoE in CIAs is conducted in real time, reflecting the genuine user experience. Strategies based on screen updates and, more directly, those that utilise the service, are the most appropriate for this purpose. It is of utmost importance to raise awareness among CIA developers about the significance of incorporating quality experience monitoring as a service feature, using monitoring APIs to export this information to third-party tools, and implementing appropriate alerting mechanisms. This monitoring should prioritise interactivity time and video quality as key indicators, without neglecting the use of other metrics for troubleshooting in the event of issues.

8. Conclusions

This survey focused on analysing methodologies and tools for the evaluation of QoE in interactive cloud-based applications. We analysed more than 28 proposals from academia and industry and identified the three most relevant human perception aspects that influence the user's QoE: visual quality, sound quality, and interactivity. We also identified the three categories of CIAs—remote desktops, cloud gaming services, and interactive web applications—and the importance of the human perception aspects in each of them. We dissected the functioning schemes of the CIAs and the particularities of each type. We also identified the components that make up the CIA architecture, how they work, and how they differ from other applications deployed in the non-interactive cloud.

Likewise, we classified the proposals in the literature based on their working strategies: screen updates, slow-motion benchmarking, measures of audiovisual degradation, instrumentation of the programming code, and indirect measures. In this process, we outlined the procedure each proposal uses to quantify QoE and identified its shortcomings. From this, we systematised the identification of the input metrics the proposals extracted from the infrastructure. We identified four main groups of input metrics and more than 21 input metrics common in the state of the art. The taxonomy revealed the tendency of researchers to opt for more than one input metric simultaneously. We further classified the proposals based on the processing they applied to the input metrics. We detected three types of processing: ad hoc, image processing, and artificial intelligence. This taxonomy revealed that artificial intelligence techniques are becoming increasingly frequent, and their results seem promising.

We grouped the output metrics into five main categories: video metrics, audio metrics, time metrics, other QoS metrics, and QoE metrics. This taxonomy revealed that a large number of proposals focused their studies on providing QoS metrics without clarifying their relationship with QoE and that much work remains to be done in this regard.

Finally, we dedicated a specific section to open issues, where we discussed possible areas for improvement in the measurement of QoE in CIAs. In particular, it is necessary to deploy strategies that can be generalised to more than one CIA and to define metrics and procedures that allow the research community to compare the accuracy and performance of the different strategies.

Author Contributions: Conceptualisation, J.A.-U. and E.M.; Data curation, J.A.-U. and E.M.; Formal analysis, J.A.-U. and E.M.; Funding acquisition, E.M., D.M. and M.I.; Investigation, J.A.-U. and E.M.; Methodology, J.A.-U., E.M., D.M. and M.I.; Project administration, E.M.; Resources, E.M.; Software, J.A.-U.; Supervision, E.M.; Validation, E.M., D.M. and M.I.; Visualisation, J.A.-U. and E.M.; Writing—original draft, J.A.-U. and E.M.; Writing—review and editing, J.A.-U., E.M., D.M. and M.I. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Spanish State Research Agency project number PID2019-104451RB-C22/AEI/10.13039/501100011033.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Markets and Markets Cloud Computing Market by Service Model (Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS)), Deployment Model (Public and Private), Organization Size, Vertical, and Region-Global Forecast to 2028. 2020. Available online: https://www.marketsandmarkets.com/Market-Reports/cloud-computing-market-234.html (accessed on 25 February 2024).
2. Xu, M.; Liu, S.; Yu, D.; Cheng, X.; Guo, S.; Yu, J. CloudChain: A Cloud Blockchain Using Shared Memory Consensus and RDMA. *IEEE Trans. Comput.* **2022**, *71*, 3242–3253. [CrossRef]
3. Ramchand, K.; Baruwal Chhetri, M.; Kowalczyk, R. Enterprise Adoption of Cloud Computing with Application Portfolio Profiling and Application Portfolio Assessment. *J. Cloud Comput.* **2021**, *10*, 1. [CrossRef]
4. Markatchev, N.; Curry, R.; Kiddle, C.; Mirtchovski, A.; Simmonds, R.; Tan, T. A Cloud-Based Interactive Application Service. In Proceedings of the 2009 Fifth IEEE International Conference on E-Science, Oxford, UK, 9–11 December 2009; pp. 102–109. [CrossRef]
5. Wang, H.; Shea, R.; Ma, X.; Wang, F.; Liu, J. On Design and Performance of Cloud-Based Distributed Interactive Applications. In Proceedings of the 2014 IEEE 22nd International Conference on Network Protocols, Raleigh, NC, USA, 21–24 October 2014; pp. 37–46. [CrossRef]
6. Menychtas, A.; Kyriazis, D.; Gogouvitis, S.; Oberle, K.; Voith, T.; Galizo, G.; Berger, S.; Oliveros, E.; Boniface, M. A Cloud Platform for Real-Time Interactive Applications. In Proceedings of the CLOSER 2011, Noordwijkerhout, The Netherlands, 7–9 May 2011; p. 7.
7. International Telecommunication Union. *Recommendation P.10/G.100: Definition of Quality of Experience (QoE)*; International Telecommunication Union: Geneva, Switzerland, 2007.
8. Laghari, K.U.R.; Connelly, K. Toward Total Quality of Experience: A QoE Model in a Communication Ecosystem. *IEEE Commun. Mag.* **2012**, *50*, 58–65. [CrossRef]
9. Arellano-Uson, J.; Magaña, E.; Morató, D.; Izal, M. Protocol-Agnostic Method for Monitoring Interactivity Time in Remote Desktop Services. *Multimed. Tools Appl.* **2021**, *80*, 19107–19135. [CrossRef]
10. Safaei, F.; Boustead, P.; Nguyen, C.; Brun, J.; Dowlatshahi, M. Latency-Driven Distribution: Infrastructure Needs of Participatory Entertainment Applications. *IEEE Commun. Mag.* **2005**, *43*, 106–112. [CrossRef]
11. Song, F.; Ma, Y.; You, I.; Zhang, H. Smart Collaborative Evolvement for Virtual Group Creation in Customized Industrial IoT. *IEEE Trans. Netw. Sci. Eng.* **2023**, *10*, 2514–2524. [CrossRef]
12. Boronat, F.; Montagud, M.; Salvador, P.; Pastor, J. Wersync: A Web Platform for Synchronized Social Viewing Enabling Interaction and Collaboration. *J. Netw. Comput. Appl.* **2021**, *175*, 102939. [CrossRef]
13. Bartik, A.; Cullen, Z.; Glaeser, E.L.; Luca, M.; Stanton, C. What Jobs Are Being Done at Home During the COVID-19 Crisis? Evidence from Firm-Level Surveys. *SSRN Electron. J.* **2020**, w27422. [CrossRef]
14. Cho, J.; Kim, S.; Kim, N.; Kang, S. Development of a Remote Collaboration System for Interactive Communication with Building Information Model in Mixed Reality. *Appl. Sci.* **2022**, *12*, 8738. [CrossRef]
15. Lee, K.; Shin, J.; Kwon, S.; Cho, C.S.; Chung, S. BIM Environment Based Virtual Desktop Infrastructure (VDI) Resource Optimization System for Small to Medium-Sized Architectural Design Firms. *Appl. Sci.* **2021**, *11*, 6160. [CrossRef]
16. Rodríguez Lera, F.J.; Fernández González, D.; Martín Rico, F.; Guerrero-Higueras, Á.M.; Conde, M.Á. Measuring Students Acceptance and Usability of a Cloud Virtual Desktop Solution for a Programming Course. *Appl. Sci.* **2021**, *11*, 7157. [CrossRef]
17. Liang, B.; Gregory, M.A.; Li, S. Multi-Access Edge Computing Fundamentals, Services, Enablers and Challenges: A Complete Survey. *J. Netw. Comput. Appl.* **2022**, *199*, 103308. [CrossRef]
18. Ralph, P.; Baltes, S.; Adisaputri, G.; Torkar, R.; Kovalenko, V.; Kalinowski, M.; Novielli, N.; Yoo, S.; Devroey, X.; Tan, X.; et al. Pandemic Programming: How COVID-19 Affects Software Developers and How Their Organizations Can Help. *Empir. Softw. Eng.* **2020**, *25*, 4927–4961. [CrossRef] [PubMed]
19. Bakaç, C.; Zyberaj, J.; Barela, J.C. Predicting Employee Telecommuting Preferences and Job Outcomes amid COVID-19 Pandemic: A Latent Profile Analysis. *Curr. Psychol.* **2023**, *42*, 8680–8695. [CrossRef]
20. Barakabitze, A.A.; Barman, N.; Ahmad, A.; Zadtootaghaj, S.; Sun, L.; Martini, M.G.; Atzori, L. QoE Management of Multimedia Streaming Services in Future Networks: A Tutorial and Survey. *IEEE Commun. Surv. Tutor.* **2020**, *22*, 526–565. [CrossRef]
21. Shakarami, A.; Ghobaei-Arani, M.; Shahidinejad, A. A Survey on the Computation Offloading Approaches in Mobile Edge Computing: A Machine Learning-Based Perspective. *Comput. Netw.* **2020**, *182*, 107496. [CrossRef]
22. Laghari, A.A.; Zhang, X.; Shaikh, Z.A.; Khan, A.; Estrela, V.V.; Izadi, S. A Review on Quality of Experience (QoE) in Cloud Computing. *J. Reliab. Intell. Environ.* **2023**, 1–15. [CrossRef]
23. Skorin-Kapov, L.; Varela, M.; Hoßfeld, T.; Chen, K.T. A Survey of Emerging Concepts and Challenges for QoE Management of Multimedia Services. *ACM Trans. Multimed. Comput. Commun. Appl.* **2018**, *14*, 1–29. [CrossRef]

24. Min, X.; Zhai, G.; Zhou, J.; Farias, M.C.Q.; Bovik, A.C. Study of Subjective and Objective Quality Assessment of Audio-Visual Signals. *IEEE Trans. Image Process.* **2020**, *29*, 6054–6068. [CrossRef]
25. Cai, W.; Shea, R.; Huang, C.Y.; Chen, K.T.; Liu, J.; Leung, V.C.M.; Hsu, C.H. A Survey on Cloud Gaming: Future of Computer Games. *IEEE Access* **2016**, *4*, 7605–7620. [CrossRef]
26. Laghari, A.A.; He, H.; Memon, K.A.; Laghari, R.A.; Halepoto, I.A.; Khan, A. Quality of Experience (QoE) in Cloud Gaming Models: A Review. *Multiagent Grid Syst.* **2019**, *15*, 289–304. [CrossRef]
27. Shi, S.; Hsu, C.H. A Survey of Interactive Remote Rendering Systems. *ACM Comput. Surv.* **2015**, *47*, 1–29. [CrossRef]
28. Metzger, F.; Geisler, S.; Grigorjew, A.; Loh, F.; Moldovan, C.; Seufert, M.; Hosfeld, T. An Introduction to Online Video Game QoS and QoE Influencing Factors. *IEEE Commun. Surv. Tutor.* **2022**, *24*, 1894–1925. [CrossRef]
29. Abdallah, M.; Griwodz, C.; Chen, K.T.; Simon, G.; Wang, P.C.; Hsu, C.H. Delay-Sensitive Video Computing in the Cloud: A Survey. *ACM Trans. Multimed. Comput. Commun. Appl.* **2018**, *14*, 1–29. [CrossRef]
30. Tang, W.; Nguyen, T.D.; Huh, E.N. A Survey Study on QoE Perspective of Mobile Cloud Computing. In Proceedings of the 2014 International Conference on Information Science & Applications (ICISA), Seoul, Republic of Korea, 6–9 May 2014; pp. 1–4. [CrossRef]
31. Casas, P.; Schatz, R. Quality of Experience in Cloud Services: Survey and Measurements. *Comput. Netw.* **2014**, *68*, 149–165. [CrossRef]
32. International Telecommunication Union. *Recommendation ITU P.911, Subjective Audiovisual Quality Assessment Methods for Multimedia Application*; International Telecommunication Union: Geneva, Switzerland, 1998.
33. Alali, F.; Adams, T.A.; Foley, R.W.; Kilper, D.; Williams, R.D.; Veeraraghavan, M. Methods for Objective and Subjective Evaluation of Zero-Client Computing. *IEEE Access* **2019**, *7*, 94569–94582. [CrossRef]
34. Kumar, R.; Yadav, A.K.; Verma, H.N. An Analysis of Approaches for Desktop Virtualization and Challenges. *Int. J. Sci. Res. Sci. Technol.* **2021**, *7*, 600–612. [CrossRef]
35. Liu, T.; He, S.; Huang, S.; Tsang, D.; Tang, L.; Mars, J.; Wang, W. A Benchmarking Framework for Interactive 3D Applications in the Cloud. In Proceedings of the 2020 53rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO), Athens, Greece, 17–21 October 2020; pp. 881–894. [CrossRef]
36. Jahromi, H.Z.; Delaney, D.T.; Hines, A. Beyond First Impressions: Estimating Quality of Experience for Interactive Web Applications. *IEEE Access* **2020**, *8*, 47741–47755. [CrossRef]
37. Dong, H.; Kinfe, A.T.; Yu, J.; Liu, Q.; Kilper, D.; Williams, R.D.; Veeraraghavan, M. Towards Enabling Residential Virtual-Desktop Computing. *IEEE Trans. Cloud Comput.* **2023**, *11*, 745–762. [CrossRef]
38. Song, T.; Wang, J.; Wu, J.; Ma, R.; Liang, A.; Gu, T.; Qi, Z. FastDesk: A Remote Desktop Virtualization System for Multi-Tenant. *Future Gener. Comput. Syst.* **2018**, *81*, 478–491. [CrossRef]
39. Hsu, C.F.; Huang, C.Y.; Liu, X. Measuring Objective Visual Quality of Real-time Communication Systems in the Wild. In Proceedings of the 2021 IEEE Seventh International Conference on Multimedia Big Data (BigMM), Taichung, Taiwan, 15–17 November 2021; pp. 9–16. [CrossRef]
40. Saverimoutou, A.; Mathieu, B.; Vaton, S. Web View: Measuring & Monitoring Representative Information on Websites. In Proceedings of the 2019 22nd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN), Paris, France, 19–21 February 2019; pp. 133–138. [CrossRef]
41. Li, W.; Sheng, J.; Yan, Y.; Zhang, S.; Deng, X.; Huang, W. The Optimization of Network Performance Evaluation Method for Virtual Desktop QoE Based on SPICE. In *Smart City and Informatization*; Wang, G., El Saddik, A., Lai, X., Martinez Perez, G., Choo, K.K.R., Eds.; Springer: Singapore, 2019; Volume 1122, pp. 141–151. [CrossRef]
42. Magaña, E.; Sesma, I.; Morató, D.; Izal, M. Remote Access Protocols for Desktop-as-a-Service Solutions. *PLoS ONE* **2019**, *14*, e0207512. [CrossRef]
43. Shu, S.; Nahrstedt, K.; Campbell, R. Distortion over Latency: Novel Metric for Measuring Interactive Performance in Remote Rendering Systems. In Proceedings of the 2011 IEEE International Conference on Multimedia and Expo, Barcelona, Spain, 11–15 July 2011; pp. 1–6. [CrossRef]
44. Hossfeld, T.; Metzger, F.; Rossi, D. Speed Index: Relating the Industrial Standard for User Perceived Web Performance to Web QoE. In Proceedings of the 2018 Tenth International Conference on Quality of Multimedia Experience (QoMEX), Cagliari, Italy, 29 May–1 June 2018; pp. 1–6. [CrossRef]
45. Exoprise. CloudReady RDP Sensor. 2021. Available online: https://www.exoprise.com/2017/01/10/monitor-remote-desktop-performance/ (accessed on 25 February 2024).
46. Wang, P.C.; Ellis, A.I.; Hart, J.C.; Hsu, C.H. Optimizing Next-Generation Cloud Gaming Platforms with Planar Map Streaming and Distributed Rendering. In Proceedings of the 2017 15th Annual Workshop on Network and Systems Support for Games (NetGames), Taipei, Taiwan, 22–23 June 2017; pp. 1–6.
47. Wehner, N.; Wassermann, S.; Seufert, M.; Casas, P. Improving Web QoE Monitoring for Encrypted Network Traffic through Time Series Modeling. In Proceedings of the 2nd Workshop on AI in Networks and Distributed Systems (WAIN), Milan, Italy, 2–6 November 2020; p. 5.
48. Graff, P.; Marchal, X.; Cholez, T.; Tuffin, S.; Mathieu, B.; Festor, O. An Analysis of Cloud Gaming Platforms Behavior under Different Network Constraints. In Proceedings of the 2021 17th International Conference on Network and Service Management (CNSM), Izmir, Turkey, 25–29 October 2021; pp. 551–557. [CrossRef]

49. Nieh, J.; Yang, S.J.; Novik, N. Measuring Thin-Client Performance Using Slow-Motion Benchmarking. *ACM Trans. Comput. Syst.* **2003**, *21*, 87–115. [CrossRef]
50. Peñaherrera-Pulla, O.S.; Baena, C.; Fortes, S.; Baena, E.; Barco, R. Measuring Key Quality Indicators in Cloud Gaming: Framework and Assessment over Wireless Networks. *Sensors* **2021**, *21*, 1387. [CrossRef] [PubMed]
51. Nguyen, T.; Calyam, P.; Antequera, R.B. Benchmarking in Virtual Desktops for End-to-End Performance Traceability. In Proceedings of the 2015 IFIP/IEEE International Symposium on Integrated Network Management (IM), Ottawa, ON, Canada, 11–15 May 2015; pp. 1268–1273. [CrossRef]
52. Varghese, D.; Saxena, M.; Sharma, A.; Ginetti, A.J.M.G. System and Method for Automated Testing of User Interface Software for Visual Responsiveness. U.S. Patent 10,114,733, 30 October 2018.
53. Laghari, A.A.; He, H.; Khan, A.; Kumar, N.; Kharel, R. Quality of Experience Framework for Cloud Computing (QoC). *IEEE Access* **2018**, *6*, 64876–64890. [CrossRef]
54. Mahmud, R.; Srirama, S.N.; Ramamohanarao, K.; Buyya, R. Quality of Experience (QoE)-Aware Placement of Applications in Fog Computing Environments. *J. Parallel Distrib. Comput.* **2019**, *132*, 190–203. [CrossRef]
55. Video Signal Input Lag Tester. Leo Bodnar, Simulator Electronics. 2021. Available online: http://www.leobodnar.com/shop/index.php?main_page=product_info&cPath=89&products_id=212 (accessed on 25 February 2024).
56. 4K HDMI Video Signal Lag Tester. Leo Bodnar, Simulator Electronics. 2021. Available online: http://www.leobodnar.com/shop/index.php?main_page=product_info&cPath=89&products_id=317 (accessed on 25 February 2024).
57. Johnsen, M. How to Measure User's Graphical Experience. 2019. Available online: https://www.brianmadden.com/geekout365/player/5714564652001 (accessed on 25 February 2024).
58. LDAT (Latency Display Analysis Tool). Available online: https://www.nvidia.com/en-us/geforce/news/nvidia-reviewer-toolkit/ (accessed on 25 February 2024).
59. TeamViewer Website. Available online: https://www.teamviewer.com/es/ (accessed on 25 February 2024).
60. Google Remote Desktop. Available online: https://remotedesktop.google.com/?pli=1 (accessed on 25 February 2024).
61. Windows Remote Desktop Client Website. Available online: https://docs.microsoft.com/en-us/windows-server/remote/remote-desktop-services/clients/remote-desktop-clients (accessed on 25 February 2024).
62. VMware View. Available online: https://www.vmware.com/products/horizon.html (accessed on 25 February 2024).
63. Citrix-All in One Workspace Solution for Secure Access. Available online: https://www.citrix.com/ (accessed on 25 February 2024).
64. Amazon Workspaces. Available online: https://aws.amazon.com/es/workspaces/ (accessed on 25 February 2024).
65. Teradici, H.P. Hybrid Work Report 2022: The Strategic Role of Digital Workspaces. Available online: https://reinvent.hp.com/hybrid-work-report-2022 (accessed on 25 February 2024).
66. Teradici, H.P. Remote Work 2020 Report—The Separation of Work and Place. Available online: https://connect.teradici.com/remote-work-2020 (accessed on 25 February 2024).
67. Dennison, K. Forbes-How The Flexible & Remote Work Debate Will Carry Into 2024. Available online: https://www.forbes.com/sites/karadennison/2024/01/24/how-the-flexible--remote-work-debate-will-carry-into-2024/ (accessed on 25 February 2024).
68. Vena Solutions-Remote Work Statistics and Trends for 2024. Available online: https://www.venasolutions.com/blog/remote-work-statistics (accessed on 25 February 2024).
69. SunRay-Clients. Available online: https://docs.oracle.com/cd/E35310_01/E35309/html/DesktopClients.html (accessed on 25 February 2024).
70. Laplink® Website. Available online: https://web.laplink.com/ (accessed on 25 February 2024).
71. Real VNC Website. Available online: https://www.realvnc.com/es/connect/download/viewer/ (accessed on 25 February 2024).
72. NVIDIA GeForce Website. Available online: https://www.nvidia.com/es-es/geforce-now/ (accessed on 25 February 2024).
73. Amazon Luna Website. Available online: https://luna.amazon.com/ (accessed on 25 February 2024).
74. PlayStation. PS Now Website. Available online: https://www.playstation.com/es-es/ps-now/ (accessed on 25 February 2024).
75. Xbox Cloud Gaming. Available online: https://www.xbox.com/play (accessed on 25 February 2024).
76. Markets and Markets, Cloud Gaming Market by Offering (Infrastructure, Gaming Platform Services), Device Type (Smartphones, Tablets, Gaming Consoles, PCs & Laptops, Smart TVs, HMDs), Solution (Video Streaming, File Streaming), Gamer Type, Region–2023 to 2028. 2021. Available online: https://www.marketsandmarkets.com/Market-Reports/cloud-gaming-market-62740366.html (accessed on 25 February 2024).
77. Illahi, G.K.; Gemert, T.V.; Siekkinen, M.; Masala, E.; Oulasvirta, A.; Ylä-Jääski, A. Cloud Gaming with Foveated Video Encoding. *ACM Trans. Multimed. Comput. Commun. Appl.* **2020**, *16*, 1–24. [CrossRef]
78. Liu, Y.; Dey, S.; Lu, Y. Enhancing Video Encoding for Cloud Gaming Using Rendering Information. *IEEE Trans. Circuits Syst. Video Technol.* **2015**, *25*, 1960–1974. [CrossRef]
79. Zhao, J.; Wang, Y.; Cao, Y.; Guo, M.; Huang, X.; Zhang, R.; Dou, X.; Niu, X.; Cui, Y.; Wang, J. The Fusion Strategy of 2D and 3D Information Based on Deep Learning: A Review. *Remote Sens.* **2021**, *13*, 4029. [CrossRef]
80. Google Docs. Available online: https://www.google.es/intl/es/docs/about/ (accessed on 25 February 2024).
81. Microsoft Office. Available online: https://www.office.com (accessed on 25 February 2024).
82. Van Steen, M.; Tanenbaum, A.S. *Distributed Systems*; Maarten van Steen Leiden: Delft, The Netherlands, 2017.

83. Marinescu, D.C. *Cloud Computing: Theory and Practice*; Morgan Kaufmann: San Francisco, CA, USA, 2022.
84. Möller, S. *Assessment and Prediction of Speech Quality in Telecommunications*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2000.
85. Chen, X.; Zhou, Y.; Yang, L.; Lv, L. User Satisfaction Oriented Resource Allocation for Fog Computing: A Mixed-Task Paradigm. *IEEE Trans. Commun.* **2020**, *68*, 6470–6482. [CrossRef]
86. Hossfeld, T.; Schatz, R.; Egger, S. SOS: The MOS Is Not Enough! In Proceedings of the 2011 Third International Workshop on Quality of Multimedia Experience, Mechelen, Belgium, 7–9 September 2011; pp. 131–136. [CrossRef]
87. International Telecommunication Union. *Mean Opinion Score (MOS) Terminology, ITU-T Recommendation P. 800*; International Telecommunication Union: Geneva, Switzerland, 1 March 2003.
88. Hosfeld, T.; Heegaard, P.E.; Varela, M.; Skorin-Kapov, L.; Fiedler, M. From QoS Distributions to QoE Distributions: A System's Perspective. In Proceedings of the 2020 6th IEEE Conference on Network Softwarization (NetSoft), Ghent, Belgium, 29 June– 3 July 2020; pp. 51–56. [CrossRef]
89. Karim, S.; He, H.; Laghari, A.A.; Magsi, A.H.; Laghari, R.A. Quality of Service (QoS): Measurements of Image Formats in Social Cloud Computing. *Multimed. Tools Appl.* **2021**, *80*, 4507–4532. [CrossRef]
90. Wang, Z.; Bovik, A.; Sheikh, H.; Simoncelli, E. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Trans. Image Process.* **2004**, *13*, 600–612. [CrossRef]
91. Setiadi, D.R.I.M. PSNR vs SSIM: Imperceptibility Quality Assessment for Image Steganography. *Multimed. Tools Appl.* **2021**, *80*, 8423–8444. [CrossRef]
92. J.247: Objective Perceptual Multimedia Video Quality Measurement in the Presence of a Full Reference. Available online: https://www.itu.int/rec/T-REC-J.247/en (accessed on 25 February 2024).
93. P.863: Perceptual Objective Listening Quality Prediction. Available online: https://www.itu.int/rec/T-REC-P.863 (accessed on 25 February 2024).
94. Farias, F.; Coelho, R. Blind Adaptive Mask to Improve Intelligibility of Non-Stationary Noisy Speech. *IEEE Signal Process. Lett.* **2021**, *28*, 1170–1174. [CrossRef]
95. Alshathri, S.; Hemdan, E.E.D. An Efficient Audio Watermarking Scheme with Scrambled Medical Images for Secure Medical Internet of Things Systems. *Multimed. Tools Appl.* **2023**, *82*, 20177–20195. [CrossRef]
96. Torcoli, M.; Kastner, T.; Herre, J. Objective Measures of Perceptual Audio Quality Reviewed: An Evaluation of Their Application Domain Dependence. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2021**, *29*, 1530–1541. [CrossRef]
97. TurboVNC | Main/TurboVNC. Available online: https://turbovnc.org (accessed on 25 February 2024).
98. Woo, M.; Neider, J.; Davis, T.; Shreiner, D. *OpenGL Programming Guide: The Official Guide to Learning OpenGL*; Addison-Wesley Longman Publishing Co., Inc.: Petaluma, CA, USA, 1999. Available online: https://www.opengl.org/ (accessed on 25 February 2024).
99. Saleme, E.B.; Covaci, A.; Assres, G.; Comsa, I.S.; Trestian, R.; Santos, C.A.; Ghinea, G. The Influence of Human Factors on 360○ Mulsemedia QoE. *Int. J. -Hum.-Comput. Stud.* **2021**, *146*, 102550. [CrossRef]
100. Nashaat, H.; Ahmed, E.; Rizk, R. IoT Application Placement Algorithm Based on Multi-Dimensional QoE Prioritization Model in Fog Computing Environment. *IEEE Access* **2020**, *8*, 111253–111264. [CrossRef]
101. Jalil Piran, M.; Pham, Q.V.; Islam, S.R.; Cho, S.; Bae, B.; Suh, D.Y.; Han, Z. Multimedia Communication over Cognitive Radio Networks from QoS/QoE Perspective: A Comprehensive Survey. *J. Netw. Comput. Appl.* **2020**, *172*, 102759. [CrossRef]
102. International Telecommunication Union, Methods for Subjective Determination of Transmission Quality, ITU-T Rec. P.800. 1996. Available online: https://www.itu.int/rec/T-REC-P.800-199608-I (accessed on 25 February 2024).
103. International Telecommunication Union, Estimating End-to-End Performance in IP Networks for Data Applications, ITU-T Rec. G.1030. 2005. Available online: https://www.itu.int/rec/T-REC-G.1030/en (accessed on 25 February 2024).
104. International Telecommunication Union, Subjective Video Quality Assessment Methods for Multimedia Applications, ITU-T Rec. P.910. 2008. Available online: https://www.itu.int/rec/T-REC-P.910 (accessed on 25 February 2024).
105. Bouraqia, K.; Sabir, E.; Sadik, M.; Ladid, L. Quality of Experience for Streaming Services: Measurements, Challenges and Insights. *IEEE Access* **2020**, *8*, 13341–13361. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.